A Novel Multi-View Table Tennis Umpiring Framework

Thesis

A NOVEL MULTI-VIEW TENNIS TABLE UMPIRING FRAMEWORK

Hnin Ohnmar Myint

A thesis submitted in partial fulfilment of

the requirements for the degree of Doctor of Philosophy

Department of Computing and Communications

Faculty of Mathematics, Computing & Technology

The Open University

United Kingdom

2018

Statement of Originality

This is to certify that to the best of my knowledge and belief, all materials presented in this thesis is the original work of the author unless otherwise stated. The content of this thesis has not been previously submitted as a part of any academic qualification to any other university or institution.

Hnin Ohnmar Myint

June 2018

Abstract

This research investigates the development of a low-cost multi-view umpiring framework, as an alternative to the current expensive systems that are almost exclusively restricted to elite professional sports. Table tennis has been selected as the testbed because, while automating the process is challenging, it has many different complex match elements including the service, return and rallies, which are governed by a strict set of regulations. The focus is mainly on the rally element rather than the whole match. Ball detection and tracking in video frames are undertaken to determine reliably the ball position relative to key reference objects like the table surface and net, and the ball's flight path is used to determine the rally's status.

While a low-cost option has benefits, it is technically challenging due to the limited number of cameras and generally low video resolution used. This thesis presents a portable multi-view umpiring framework that identifies each state change in a rally. It makes three significant contributions to knowledge: i) a reliable ball detection strategy that accurately detects the location of the ball in low-resolution sequences; ii) a novel framework for ball tracking using a multi-view system, and iii) a new state-machine based evaluation system for analysing table tennis rallies.

In a series of ten different test scenarios, the system achieved an average of 94% system detection rate and 100% accurate decisions. A test sequence of duration 1 s can be processed in 8 s, leading to a delay of only 7 s, which is considered acceptable for practical purposes. This solution has the potential to reform the way matches are umpired, providing objectivity in resolving disputed decisions. It affords an economic technology for amateur players, while the multi-view facility is extendible to other relevant ball-based sports. Finally, the ball flight path analysis mechanism can be a valuable training tool for skills development.

Declaration

The work presented in this thesis is an original contribution of the author. Parts of this thesis have appeared in the following:

Peer-Reviewed Publications:

1. **Myint, Hnin;** Wong, Patrick; Dooley, Laurence, and Hopgood, Adrian (2016). Tracking a table tennis ball for umpiring purposes using a multi-agent system. In: *The 20th International Conference on Image Processing, Computer Vision, & Pattern Recognition*, 25-28 Jul 2016, Las Vegas, USA.
2. **Myint, Hnin;** Wong, Patrick; Dooley, Laurence, and Hopgood, Adrian (2015). Tracking a table tennis ball for umpiring purposes. In: *Fourteenth IAPR International Conference on Machine Vision Applications (MVA2015)*, 18-22 May 2015, Japan.

Poster Presentations:

Myint, Hnin (2014). Multi-agent system for real-time video umpiring. In *IET Midlands Power Network Engineering Event (in association with industry)*, Coventry University, UK, Mar 18, 2014.

Poster Awards:

Myint, Hnin (2013). Multi-agent system for real-time video umpiring. In *Postgraduate Student Poster Competition, The Open University*, UK, July 18, 2013.

Dedication

My respected parents and a very supportive husband.

Acknowledgments

At this moment, I would like to express my gratitude to some important people who have always been a cornerstone for encouraging me to pursue this PhD research degree. This thesis could not have been produced without the inspirations, encouragement, and support from many people, to whom I am greatly indebted.

First and foremost, I would like to express my earnest tribute to my respected parents. To see a daughter holding a Ph.D. degree from the UK is a lifelong dream of my parents and I would never miss my parents at any point in my life. Their unconditional and immeasurable sacrifices have empowered me to reach this stage and successfully complete this thesis. With the deepest gratitude, I dedicate this thesis to them.

My sincere gratitude goes to my principal supervisor, Dr. Patrick Wong who was very generous to provide his valuable time for insightful technical advice, fruitful discussions and guiding me throughout this research. I am also indebted to my co-supervisors Prof. Laurence S. Dooley and Professor Adrian Hopgood for always providing me with many useful comments, valuable insights and excellent guidance. Their inspiring instructions and recommendations pointed me in the right direction. This work is in its current shape only through their continuous supports, insistence, and attention to detail.

I also would like to acknowledge the valuable advice from Dr. Adrian S. Poulton. His constructive feedbacks and effective suggestions have improved the quality of this thesis. I would like to express my special thanks to Dr. Soraya Kouadri Mostéfaoui for her valuable advice and constant support throughout this entire

research. Her encouraging words helped me immensely and tremendously contributed towards the production of this thesis.

Furthermore, I would like to thank all my friends and members of the *neXt Generation Multimedia Technologies* (XGMT) Group for their warm friendship and helpfulness. I am honourable blessed by the generous financial support of the Research School at the Open University for granting me the opportunity and providing me the right facilities to engage this research degree. This research/work was supported by the Open University which is incorporated by Royal Charter (RC 000391), an exempt charity in England & Wales and a charity registered in Scotland (SC 038302). The Open University is authorised and regulated by the Financial Conduct Authority. Thanks to all the administrative staff in the department and the research school with special appreciation for Sue, Donna, Jennifer and Lyn.

Last, but by no means the least, I would like to extend my appreciation to my husband, Lon Chanborey for his innumerable sacrifices, endless support and encouragement throughout this PhD journey.

Table of Contents

List of Figures

List of Tables

List of Abbreviation

3D	Three-Dimensional
2D	Two-Dimensional
ACL	Agent Communication Language
ACTMD	Adaptive Colour Thresholding and Motion Detection
AI	Artificial Intelligent
ANN	Artificial Neural Network
AOP	Agent-Oriented Programming
BDA	Ball Detection Agent
BPNN	Back-Propagation Neural Network
BS	Background Subtraction
CI	Computational Intelligence
CT	Colour Thresholding
EPD	Energy Peak Detection
FBD	Feature-based Ball Detection
FDA	Feature Detection Agent
FIFA	Federation Internationale de Football Association
FIPA	Foundation for Intelligent Physical Agents
fps	Frames Per Second
FTF	Face-To-Face
GAs	Genetic Algorithms
GLT	Goal-Line Technology
GUI	Graphical User Interface
HSV	Hue-Saturation-Value
IDE	Integrated Development Environments
IFAB	International Football Association Board
ITTF	International Table Tennis Federation
IVSC	Inter-View Self-Correction
JADE	Java Agent Development Framework
KBS	Knowledge-Based Systems

KF	Kalman Filter
KQML	Knowledge Query and Manipulation Language
LQE	Linear Quadratic Estimation
MAS	Multi-Agent System
MDA	Motion Detection Algorithm
MVCA	Multi-View Correction Agent
NPO	Net Pole Origin
OMLB	Object That Is Most Likely the Ball
OOI	Object-Of-Interest
OOP	Object-Oriented Programming
OR	Overlapping Region
OUTTDB	The Open University Table Tennis Database
PC	Personal Computer
PNP	Perspective-N-Point
PTZ	Pan-Tilt-Zoom
RANSAC	Random Sample Consensus
RGB	Red-Green-Blue
RMSE	Root Mean Square Error
ROI	Regions of Interest
RPM	Revolutions Per Minute
SBS	Side-By-Side
SDA	State Detection Agent
SOMM	Second Order Motion Model
SW	Sliding Window
TCA	Trajectory Construction Agent
TPT	Two-Pass Thresholding
UR	Undetectable Region
VRL	Visible Region Left
VRR	Visible Region Right

List of Symbols

a	Half of the Screen Width
$x1, x2$	Variable
x^l, x^r	Horizontal Positions of the Points in the Left and Right Images
c_x^{left}, c_x^{right}	Principal Points
O	Centre of Projection
T	Total Length Bet Two Cameras: cm
f	Focal Length of the Camera.
d	Disparity
Z	Depth
X	Value of Horizontal 3D Position of the Ball (Distance)
x_1, x_2	Screen Coordinate of the Centre of the Ball in Pixels
W	Screen Width
Z_1, Z_2	Associated Distance Between the Ball and Each Camera
Y	Value of the 3D Ball Position in the FTF Configuration
H	Screen Height
R_i	Pixel Values
μ_i, σ_i	Mean and Standard Deviation
m_i	User Defined Multiple (Tolerance)
t	Threshold Margin
i	Index of the Colour Channels
$I_b(x, y)$	Value of Pixel (x, y) in the Binary Image
I_T	Threshold for Binarisation

$I_{fd}(x, y)$	Value of Pixel (x, y) which is the Result of Frame Differencing
Δt	Time Difference
a_n	Ball's Acceleration at Frame n
C_1	Centre of the Ball in the First Frame
B_{n-1}	Detected Ball Location in the Previous Frame
v_n	Velocity at Frame n
P_{n+1}	Predicted Ball Location of the $n + 1^{th}$
S_n	Total Number of Detection Score of the n^{th} Frame
$dist[DP]$	Less Than the Predefined Distance
max_d	The Point will be Awarded
r_D	Radius of Detected Ball
min_r	Predefined Minimum Radius
max_r	Maximum Acceptable Radius
$Avg\_RMSE\_HSV_D$	Average RMSE between the Detected Ball's HSV Colour
$acceptable\_HSV$	Acceptable Threshold
$\boldsymbol{E}$ *(x, y, z)*	3-Dimensional Error Vector
a_n, b_n, c_n, d_n, e_n, f_n, g_n, h_n, i_n *and* j_n	Coefficients of the Quadratic Surfaces

Chapter 1

Introduction

1.1 Overview

In the past decades, the application of computer vision which can extract high-level understanding from digital images or videos has grown dramatically. Due to its rich potential in many different application areas, it has been utilized in pattern recognition, medical imaging, military affairs, security surveillance, three-dimensional (3D) reconstruction and robotic applications. The development of computer vision has made it feasible to detect and track a target object. Many sports are increasingly considering this technology for verification purposes in key umpiring decisions because one of the main controversies surrounding sport is an incorrect decision made by match officials (Fowler, 2012).

To assist a human umpire in making a more accurate decision, there has recently been a growing trend to employ vision-based systems for reviewing problematic refereeing decisions at professional sporting events like the Wimbledon Tennis Championships and English Premier League. Likewise, researchers are exploring computer vision technologies in developing game highlight detection algorithms and virtual replay systems. Examples include determining whether a target object is inside or outside of the playground's boundary such as Hawkeye (Hawk-Eye Innovations, 2018; Owens et al., 2003),

the close-up of a player after a goal (C. O. Conaire et al., 2009) or showing the trajectory of the ball on a virtual court by rendering the graphics (McIlroy, 2008), and detecting predefined events of goals in soccer (Chen and Zhang, 2006). This technology not only supports human umpires but also provides a proof of their decision making and reduces the chance of errors.

Despite offering significant advantages, existing systems have still struggled to meet ever-growing demands in terms of high deployment costs, complex installation and current limited functionality. Moreover, they do not reach to the level of automatically umpiring the whole match. A coach or a domain expert is required to help players to focus on and interpret critical elements of recorded movement patterns (Bacic et al., 2017). As a human umpire may not always be present on site in self-training, the desire for automated analysis of sporting activities is still increasing (Mukai et al., 2011). Consequently, a cost-effective, easy-to-operate umpiring system is an attractive alternative option, with such technology potentially supporting lower-tier and amateur sporting communities.

However, selecting a type of sport for academic experimentation involves consideration of many factors and it is quite ambitious to develop a system which can umpire different types of matches. Therefore, table tennis has been selected as the testbed-sport because while automating the umpiring process is challenging, it has many different complex match elements including the service, return and rallies, which are governed by a strict set of rules. Compared with other sports such as tennis or football, table tennis is an indoor game which does not need a wide area for setting up the lab environment. While offering ease of

installation, it is a very challenging sport to umpire in terms of the number, size and speed of the objects and the required observations and regulations to be upheld. A table tennis ball is small (approximately 4 cm in diameter, 2.7 grams in weight) and blurs due to its speed (around 112.5 kilometres per hour (Table Tennis Master, 2017)). This provides a motivation to conduct this research.

1.2 Table Tennis Umpiring

Table tennis is a very fast-moving Olympic sport with millions of players worldwide. It is governed by the worldwide International Table Tennis Federation (ITTF) and the official rules are specified in the ITTF handbook (International Table Tennis Federation, 2018; Delano and L.F. (n.d.), 2018). It has many different complex match elements including the service, return, and rallies. Firstly, it is necessary to describe the usage of table tennis's terms. While not describing everything, this chapter provides some of the key terms to be used which include:

- A rally is a series of returns between the server and receiver.
- The service in table tennis is the starting point for every rally.
- The server is the player due to strike the ball first in a rally.
- The receiver is the player due to strike the ball second and provides a return in a rally.
- A fault occurs when a player fails to continue the rally.
- A point is won if the opponent makes a fault.
- A rally is completed where the result is scored.
- A game is a leg of a match consisting of a series of rallies.

- The table or playing surface is 274 cm long, 152.5 cm wide, and 76 cm high. It is uniformly matte coloured and divided into two by a net of height 15.25 cm as shown in figure 1.1.

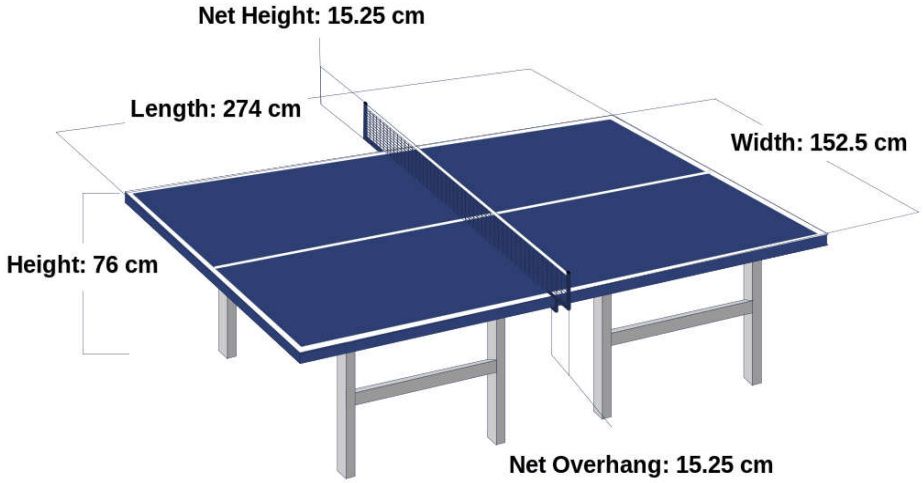


Figure 1.1: The official dimensions of table tennis table (ITTF, 2018, n.d.)

The full table tennis rules are very comprehensive. For the purpose of simplicity, the main table tennis rules considered by this thesis are summarised as below based on the rules from ITTF (International Table Tennis Federation, 2018). At the beginning of the play, service shall start with the ball resting freely on the open palm of the server's stationary free hand. The server shall then project the ball near vertically upwards, without imparting spin, so that it rises at least 16 cm after leaving the palm of the free hand and then falls without touching anything before being struck. The receiver shall then make a return and thereafter server and receiver alternately shall each make a return. The ball, having been served or returned, shall be struck so that it touches the opponent's court, either directly or after touching the net assembly. From the start of service until it is struck, the ball shall be above the level of the playing surface and behind the

server's end line, and it shall not be hidden from the receiver by the server or by anything they wear or carry. However, if the ball touches the net and does not bounce on the opponent's side of the table, the server loses the point. Moreover, various faulty conditions can occur when a player obstructs the ball such as if the ball passes over the player court or beyond his or her end line without touching his or her court, after being struck by an opponent. It is the responsibility of the umpire to check whether the play complies with the requirements of the Laws. However, some of the judgments described by the table tennis rules are very difficult for a human being to make and are a constant cause of debate and argument. To this end, a purposely built computer vision system could help evaluate these difficult observations.

1.3 Research Aim and Focus

To investigate the feasibility of an automatic umpiring system, the aim of this research is to develop a framework which can be used for umpiring tennis table rallies. The focus is on developing techniques and implementing algorithms rather than building a complete system. To make the system accessible by amateur players, the system is aimed to be low-cost, portable and simple to install. Figure 1.2 presents the proposed framework as a block diagram. It is composed of 7 main sequential processes (blocks) which are;

1. Video Capturing
2. Stereo/Multi-View Forming
3. Calibration
4. Ball Detection

5. Multi-View Tracking
6. Umpiring Rallies
7. Umpiring Match

Each individual block and its function are summarised below.

Video Capturing (Block 1): The block diagram starts with a Two-Dimensional (2D) video capturing process by acquiring data from real-world scenes in which the view of the play is captured with multiple cameras (shown as C1 .... Cn).

Stereo/Multi-View Forming (Block 2): The intention of capturing a scene with more than one camera is one camera alone cannot give depth information of an object. In an umpiring system, the 3D locations of objects of interest are essential information. In this research, the ball is the target object to detect, and the net pole and table edges are assigned as reference points/lines to make a location comparison.

Calibration (Block 3): The camera pair that is employed to detect the 3D location of the objects should be placed at the same height and well aligned. A small misalignment could lead to a large error in their 3D locations. To reduce this, a calibration process is required for reducing distortions and correcting misalignments between cameras.

Ball Detection (Block 4): Image segmentation, object detecting and tracking, and 3D position derivation are involved in this process. As the table tennis ball size is small and often travels at high speed, its features such as colour, shape, and size are all distorted and no longer apparent. This makes the ball detection

task difficult and therefore it is necessary to develop a robust mechanism to achieve reliable detection.

Multi-view Tracking (Block 5): Although 3D information can be derived from either stereo or multiple cameras, if the view can be captured with more cameras, it could reduce the risk of occlusion and provide a better chance in detection. However, it requires a strategy to track the ball among multiple cameras. Since multiple cameras will provide more data to analyse, it would also increase the system computation load. One way to solve this problem is to develop the system with multi-agent technology because agents can make many observations in parallel and can distribute the workload across the system. Each agent can work on their own task, act autonomously and make decisions based on the implemented rules. In this way, the complex umpiring process can be broken down into simpler components and the system could be more maintainable, adaptable, extendable and reusable while maintaining the integrity. Therefore, a multi-agent method is proposed in this research instead of developing with a conventional programming method.

Umpiring Rallies (Block 6): This process analyses the 3D joint trajectory. This includes detecting features such as travelling direction of the ball, its current position, and the rate of change of velocity and acceleration. Based on these features, the system can identify the current state of a rally; for example, whether the travelling ball is crossing over the net, bouncing on the table surface or over the table end line. Since the condition of the rally can be in exactly one of a finite number of states at any given time, the finite state machine (FSM) model could

be a possible solution for umpiring rallies. According to the standardised table tennis rules, the FSM can evaluate whether the rally is in the correct state or a fault.

Umpiring Match (Block 7): The ultimate goal of an automatic umpiring system is to umpire the whole match. This includes determining when a rally starts and completes, score the matches according to the rules and determine the winner. Scoring a match is difficult because there are strict rules governing who serves, when to change server, when to swap sides, the maximum time a game can take and so on. Moreover, determining the start of a rally is also challenging because many players have different pre-service actions, such as bouncing the ball on their bats, table or the floor. Furthermore, the system must be able to distinguish whether the match is being played or being stopped. However, it has been found that developing a fully automatic umpiring system is a very complex task and there is a limited time for this research. Therefore, every table tennis rule will not be implemented in this research and block 7 is regarded as beyond the scope of this thesis but can be future extensible work. Based on these considerations, the key experimental assumptions are made as follows:

- Assume the table and net position are fixed.
- During a rally, if the ball position is out of the view of all the cameras for more than 1 second, it is considered as a fault.

The focus of this thesis is especially on the rally element rather than the whole match. Thus, ball detection and tracking in video frames are undertaken to reliably determine its position relative to key reference objects such as the table

surface and net, and the ball's flight path is used to determine the rally's status. In the following figure 1.2, the blocks 1 to 6 will be covered in this research and the block 7 is for future research. The detailed processes of blocks 1 to 3 can be found in Chapter 3. The contribution chapters 4, 5 and 6 present the detailed implementation of yellow blocks 4, 5 and 6. The future work Chapter 7 represents green block 7.

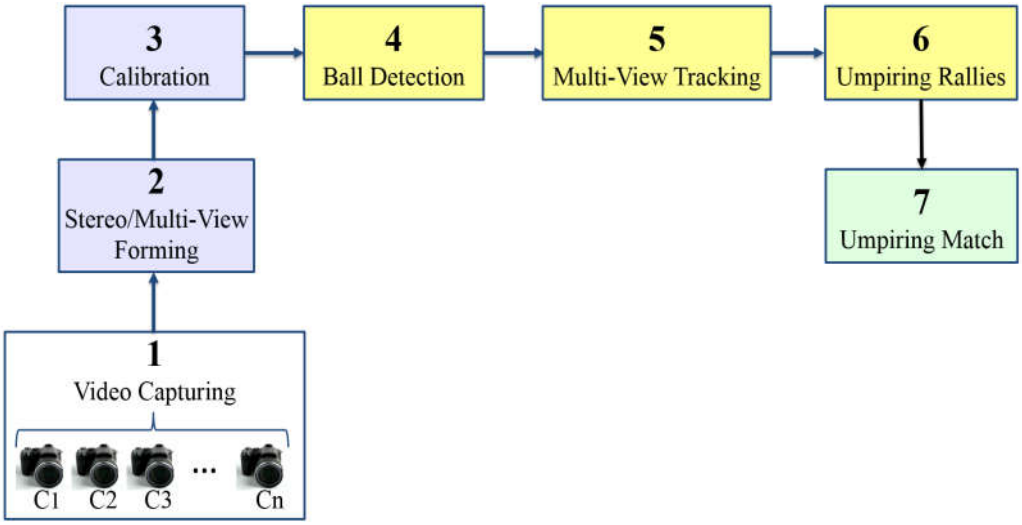


Figure 1.2: Multi-View Table Tennis Umpiring Framework

1.4 Research Challenges

While table tennis has a myriad of diverse rules governing the legality of a rally, there are many technical challenges in developing a low-cost automatic umpiring system. Some of the key issues to be addressed include:

- Tracking a table tennis ball alone is a difficult task due to its small size.
- The ball often travels at high speed and it is one of the fastest moving sports in the world (Rusdorf et al., 2007). The effect on the captured image of the ball is shape distortion, allied with object blurring and colour deviation caused

by the light level change between frames in a fast video sequence, and occlusion by the players or their bats.

- While a low-cost option has benefits, it is technically challenging due to the limited number of cameras and generally low video resolutions used.
- Umpiring a complete rally is a very complex task and requires a lot of decisions and the demand for timely and accurate observations of rallies imposed by the table tennis rules (International Table Tennis Federation, 2018) is very high. To achieve that, the analysis must be processed very quickly, and produce reliable judgments rapidly and consistently.
- Some rallies are very difficult to judge, such as; whether the ball touches the net, hits the edge of the playing surface, which is legal, or the side of the table, which is a foul. This kind of judgment is very challenging even for a professionally trained umpire (Wong and Dooley, 2010).
- Determining a fault in a rally is also challenging because many conditions can occur such as a fault due to double bounce, a fault due to a return not bouncing on the right side of the table, and a fault due to the ball hitting the floor.

The availability of advanced computer vision technologies and artificial intelligent (AI) tools which can aid in developing algorithms more efficiently motivates this challenging research. Exploring these useful technologies and developing a strategic methodology, the challenge of how to effectively identify the state of a rally which can be helpful for match umpiring, was the main motivation behind this research. This provided the context for the overarching

thesis research question and related objectives which are distilled in the next section.

1.5 Key Research Questions and Objectives

From the above discussion, the main research question was framed:

How can table tennis rallies be automatically umpired?

While the goal is to umpire table tennis rallies, one important activity of such system is to accurately and rapidly track the location of the ball during a match because the result of the ball location plays a crucial role in analysing rallies and can seriously impact the decision making. Moreover, only after the ball locations in consecutive frames are detected can the trajectories of rallies be constructed, and the legality of the rally determined. Therefore, it is necessary to develop an efficient ball detection algorithm and critically synthesise a tracking system. After that, an umpiring system can be developed for checking a large number of events happening in rallies against the rules. Examples of events are whether the ball is served behind the end line of the table or bounces on the correct ends of the table. The system must be able to check these events and provide a decision rapidly. Consequently, the following set of three research objectives was framed to underpin the above overarching question:

1. **To develop and critically evaluate an efficient ball detection algorithm.**

Justification: To umpire a match automatically, one critical requirement is to detect the ball's 3D position and compare it with the positions of other objects such as the table and net. Although the ball is small and often travels at high

speed, the proposed system is aimed to be low-cost, produce the result in real-time with high detection rate, be portable in implementation and accessible not only to professional umpires but also amateur players. Therefore, it is necessary to develop a new ball detecting algorithm.

2. **To develop and critically synthesise a multi-view ball tracking system.**

Justification: If multiple cameras each capture a specific portion of the view, it can produce higher resolution, reduce occlusion and improve detection. As each individual camera does not have to cover the entire table, the cameras can be placed closer to the objects of interest (e.g. ball and table) so that better depth resolution can be achieved when deriving their 3D positions, and the apparent sizes of the objects are also larger in the views. Furthermore, depth information can also be derived from multi-view data. Tracking the ball in the rallies by analysing the visual and depth information enables the detected ball to be projected into a 3D space. However, determining a ball's complete trajectory based on combining the results from multiple cameras increases computation load, due to synchronising and processing data from multiple cameras. Therefore, a new multi-view ball tracking strategy needs to be investigated.

3. **To develop an automatic umpiring system for critically evaluating a table tennis rally.**

Justification: After constructing a complete ball's trajectory, the system needs to critically analyse table tennis rallies by detecting various features of ball's trajectory, identify the different state of rallies, and determine the legality of play according to the rules. This involves the development of a finite state machine

and multi-agent umpiring system for handling a dedicated task and evaluating it autonomously. This is followed by rigorous testing with recorded and live videos of real matches, to identify the limitations and explore possible improvements.

1.6 Contributions

To fulfil the aforementioned objectives, this thesis presents a new portable, multi-view umpiring framework that identifies each state change in a rally, including the service start and the point when a fault occurs. It makes three original significant contributions to knowledge:

1. A reliable ball detection strategy that accurately detects the location of the ball in low-resolution sequences**.**
2. An extendable novel framework for ball tracking using a multi-view system.
3. A new state-machine based evaluation system for analysing table tennis rallies.

Although this project focuses on table tennis match umpiring, the methodologies contributed and technologies developed could be adopted in video analysis of other sports such as cricket, football, tennis, soccer, baseball, basketball, and volleyball. They are also applicable in video content retrieval and in 3D vision in inventory control when checking for ball shaped objects. Since this research aims to develop an affordable system by employing low-cost stereo vision equipment, the proposed system can be used not only by professional but also amateur players. Moreover, it has the potential to reform the way matches are umpired, providing helpful information in resolving disputed decisions. Finally, the ball

flight path analysis mechanism can be a valuable training tool for skills development.

1.7 Thesis Structure

The rest of thesis is organised as follows:

- Chapter 2 presents an intensive survey of state-of-the-art object detection and tracking techniques with a special focus on table tennis ball tracking. It also discusses the benefits and drawbacks of different methods which are being used and critically reviews each method and comments on why some techniques are suitable or unsuitable for this research and explain what they have done or what they have not done effectively, leaving a gap to fill. The critique in Chapter 2 helps to identify the most suitable research methodology.
- Chapter 3 focuses on the research methodology adopted in this thesis including building a testbed, experimenting with test sequences and evaluating the results. It also discusses the choice of development platform, the new framework's validation procedures and the key performance metrics applied to critically assess the comparative performance of all new algorithms and the proposed system.
- Chapter 4 details the first contribution, which is the development of a stereo-view ball detection strategy that accurately detects the location of a ball in low-resolution sequences. The algorithm performance is critically evaluated with real match scenes sequences from The Open University table tennis database (OUTTDB). Work from this chapter has been published in (Myint et al., 2015).

- Chapter 5 presents the second contribution which extends the stereo-view ball detection algorithm to be a multi-view ball tracking system. This provides continuous tracking and is efficient in computation. The system was tested with complete table tennis rallies, and the results of experiments show that accurate and rapid tracking can be achieved even under challenging conditions, including occlusion and colour merging. It can continuously track the ball with only a small occasional deviation. The proposed design of the multi-camera set up also enables the system to be cost-effective, portable and suitable for umpiring purposes. Work from this chapter has been published in (Myint, Hnin, 2016)
- Chapter 6 describes the novel multi-view table tennis umpiring framework. It develops a new multi-agent state-machine based evaluation system for analysing table tennis rallies. The system can identify different states of situations where a point is awarded, such as whether the ball is over the server end-line, bounces on the server side of the table or crosses over the net. Moreover, the system has an ability to declare when a fault occurs (extracted from the Table Tennis Rules (International Table Tennis Federation, 2018)) such as a fault due to double bounce, a fault due to a return not bouncing on the right side of the table, or a fault due to the ball drop under table and hitting the floor etc. The proposed system is applied to a variety of complex table tennis rallies.
- Chapter 7 discusses some potential research directions which can exploit the methods and algorithms developed in the research. This can be future research

work which considers some possible enhancements and extensions to the framework presented.

- Chapter 8 presents some conclusions on the key findings and original contributions described in this thesis.

1.8 Summary

This chapter began with an overview of computer vision technologies applied in object detecting and tracking. It was followed by explaining the potential of automatic systems for virtual replay and umpiring various sports. It went through some examples not only demonstrated in research but also in commercial achievement. Subsequently, it formulated the three principal research objectives addressed in this thesis and possible techniques to address the challenges based on evidence. A concise overview of the main research question, related objectives, the methodology to fulfil the aim, contributions made, and thesis structure have been provided. The next chapter presents a comprehensive literature survey of object tracking and their associated technologies for developing an automatic table tennis match umpiring system.

Chapter 2

A Survey of the Literature

2.1 Introduction

Over the last decade, vision-based detection technologies have gained immense attention across academia-industries such as public surveillance, intelligent transportation, industrial control, military purposes, security, automatic tracking system and many other numerous applications. Due to its usefulness and increasing demand for automated detection system, object detection and tracking became one of dominating research areas. As researchers have been making efforts to enable optimal tracking results, many developments have been done in this field. Besides the proliferation of high-powered computers and inexpensive video cameras, the availability of **AI** tools which can aid in developing algorithms more efficiently also support the rapid growth of advanced computer vision technologies.

However, the complexities of real-time constraints and expected functional characteristics often raise questions over existing algorithms and their efficacy. Although various methods have their strengths, they also have their limitations, and this motivates towards developing a more effective solution. Despite this, the existing algorithms do not always fit with individual's requirement and are not directly applicable to the intended application. Thus, it

is necessary to review the current methods, and critically analyse the benefits and drawbacks of different techniques to explore their extensibility.

This literature survey aims to understand the state-of-the-art development of automatic sport umpiring and to identify the most suitable object detection and tracking techniques, which are crucial to the success of this project. Since this research is mainly focused on tracking a table tennis ball for umpiring purposes, the first part of this chapter gives an intensive survey of object detection and tracking literature with a special focus on a fast-moving ball and similar objects. In the second part, an overview of what computer vision technologies have been employed in various sports, the current trend of virtual replay systems not only in research accomplishment but also in commercial achievement is presented. The third part of this chapter gives the revision of the state-of-the-art research work regarding technology in table tennis. The last section of this chapter discusses the challenges that are frequently faced in object detection and tracking research and provides possible solutions.

2.2 The Process of Object Tracking

According to an intensive survey (Hawk-Eye Innovations, 2018; Anuj and Krishna, 2017; Balaji and Karthikeyan, 2017; Shih, 2017; Fan et al., 2016; Reddy et al., 2015; Parekh et al., 2007; Yilmaz et al., 2006; Wang and Parameswaran, 2004), almost every tracking system requires a sequence of processes called object **Segmentation-Detection-Tracking** mechanism. Based on this, the following sections 2.2.1 to 2.2.3 will be organised as below to introduce the process of these essential mechanisms:

- **Segmentation:** the process of partitioning an image to achieve a focus area to analyse.
- **Detection:** the process of examining instances of targeted objects.
- **Tracking:** the process of locating or estimating positions of moving an object over time.

Since the aim is to build an automatic umpiring system, accurately and rapidly determining the location of the ball during a match is one important goal. This is because the result of ball location plays a crucial role in analysing rallies and can seriously impact the decision making. Therefore, it is necessary to review a variety of detecting and tracking methods and explore possible extensions.

2.2.1 Segmentation

The aim of image segmentation is to get rid of unnecessary parts and try to narrow down the detected area. The process includes partitioning the image into perceptually smaller regions which are called **Regions of Interest (ROI)**. Every segmentation algorithm addresses two problems, the criteria for a right

partition and the method for achieving efficient partitioning (Yilmaz et al., 2006). Among a large number of different approaches, the most widely reported segmentation methods are **Colour Thresholding**, **Background Subtraction** and **Frame Differencing**, and **Edge Detection.**

2.2.1.1 Colour Thresholding Methods

Colour Thresholding (CT) is widely used to segment image into background and object. Low computational cost algorithms making good use of colour thresholding whenever appropriate. Some researchers (Fitriana et al., 2016; Qazi et al., 2015; Bao et al., 2012; Chen et al., 2010, 2011; Wong and Dooley, 2010; Arenas et al., 2009) employ colour thresholding-based segmentation methods to extract a candidate ball from each video frame. In colour thresholding, pixels that have values within a threshold range are considered to be from the object if the ball has of a uniform colour such as white or orange. However, selecting the right threshold value is challenging. The threshold value could be different for different videos, and even within the same video. No exact value may correctly identify all changes between different frames. Although this method is simple, the result of CT is very sensitive to the threshold setting, and it is also difficult to set the threshold appropriately. If a threshold value is too low, it will identify shot changes that do not exist. If a threshold value is too high, it will miss some shot changes. To this end, Wong and Dooley (Wong and Dooley, 2010) proposed a **two-pass CT method**, that uses a strong (rough) threshold in the first pass to identify the locations of the candidate object which is a ball, and a slight (tolerant) threshold on the second

pass applying only to the regions defined in the first pass to recover the missing pixels filtered in the first pass. While satisfactory results were achieved, it only covered the service part of the rallies, where the ball is not traveling at high speed. Likewise, Zhang (Zhang et al., 2010) proposed a **dilation based method** known as the growth of sampled points to recover the distorted ball shape. Although their methods achieved some improvements, the performance of CT is generally poor when the ball is colour merging against a complex background. Furthermore, the images of the extracted objects are often distorted due to the light intensity of the environment which is usually uneven, and this means the colour of the object in the image varies.

2.2.1.2 Background Subtraction Methods

An alternative method is **Background Subtraction (BS)** which works by segmenting the current frame into background and foreground regions. The background can be assumed as any static or periodically moving parts of a scene which remain under certain conditions. Any significant changes in an image region from the background model signifies a moving object. In BS, building a representation of the scene is called background modelling and then deviations from that model for each incoming frame are called foreground regions. Which means any pixel which does not fit the background model can be assigned as foreground and the rest will be background. A background model can be created by a small number of initial frames, or the other way around is to select a sample of background. Once learned, it can be used for making a comparison against the current frame. The background model can also be computed once only, or it can

be incrementally updated according to the continuous frames. Recently, an **artificial neural network-based model** (Maddalena and Petrosino, 2013) has been proposed for automatically adapting the model of background, moving foreground, and stationary foreground. One of the early approaches of BS technique for automatically extracting tennis court lines and detecting the location of the tennis player can be found in Sudhir's paper (Sudhir et al., 1998). In their methods, a model of the tennis court is developed by using a priori knowledge of camera geometry, dimensions and connectivity of a tennis court. Thereafter, the model is used for developing a colour-based court line detection algorithm and motion-based player tracking algorithm. The most widely used BS techniques are the **Median Filter** (Cutler and Davis, 1998), the **Particle Filter** (Nummiaro et al., 2003) and **Gaussian Mixture Models** (Zivkovic, 2004).

If the object is identical in the whole frame, a well-developed model of the background followed by the object segmenting algorithm may be useful. However, the performance of pixel-level subtraction mostly degrades in outdoor scenes due to shadows, reflectance and repetitive object motion (Tong-yao, 2011). BS approaches can be easily affected by noise factors, such as moving crowds, cluttered background, and luminance changes. Recently, Lu, Yang and Zhao (Lu and Yang, 2017; Zhao et al., 2017) proposed BS methods that have the capabilities of modelling the changing illumination, noise, and the periodic motion of the background regions. However, their methods require reconstructing the lost region with one more technique such as inpainting to recover several holes inside the segmented object. That makes computational overhead and will not suit for real-time application.

One of the limitations of BS is the requirement of stationary cameras. Although much effort (Zhao et al., 2017; Yadav et al., 2014; Tong-yao, 2011; Bayona et al., 2010; Zivkovic, 2004; Toyama et al., 1999) have been made in some enhancements to BS techniques, camera motion usually distorts the background models. To solve this, Wu (Wu et al., 2017) proposed an adaptive background thresholding scheme with a freely moving camera such as a hand-held camera or professional **pan–tilt–zoom (PTZ)** camera. However, it has been shown that among their different types of tested sequences, segmenting football players from YouTube live soccer video sequence performs worse than the others as a target segmented object is a very small size in the video. Therefore, their technique may not be suitable for tracking a table tennis ball, which is much smaller than a player. Moreover, background modelling is computationally expensive due to the need to cope with the expansion of the scale, translation, rotation, and deformation (Chakroun et al., 2011).

2.2.1.3 Frame Differencing Methods

Instead of modelling, the simplest way to segment the background can be achieved by frame differencing based on its motion if the target object has a nature of moving. In frame differencing, the existence of a moving object is segmented by the difference between two consecutive frames. In this way, the whole part is obtained as background and the differential part is obtained as candidate objects to be analysed further. As the calculation made is easy and simple for implementation, frame differencing algorithms are widely applied in

object detecting and tracking with low complexity (Abdelli and Choi, 2017; Srivastav et al., 2017; Tanaka and Miura, 2017; Sengar and Mukhopadhyay, 2016; Yilmaz et al., 2006).

While this method is simple and useful, it will work only if the surrounding area of the ball remains unchanged and the ball is in motion across consecutive frames. When there are multiple motions or no motion between two or several successive frames, poor segmentation results are achieved. If the object features are not much different between consecutive frames, it usually provides incomplete object regions and spillage into several regions. To solve this, Zhang (Zhang et al., 2010) proposed the growth of sampled points method which is aimed to recover pixels lost unintentionally during the **adjacent frame differencing**. Although their experimental results suggested promising detection results, pixels belonging to the background could sometimes be incorrectly classified as belonging to the detected object (ball) and that could make the detected object's size inaccurate.

2.2.1.4 Edge Detection Methods

An alternative method for image segmentation is edge detection, which includes segmenting the image by locating the sharp edges which are discontinuous. These discontinuities bring changes in pixels intensities which define the boundaries of objects and can extract a set of curved line segments termed edges. In this method, object segmentation is achieved by evolving a closed contour to the object's boundary, such that the contour tightly encloses the object region. Example of edge detection algorithms include **Canny** (Canny,

1986), Laplacian (Chen et al., 2012), **Prewitt** (Seif et al., 2010), **Sobel** (Ma et al., 2010), **Robert** (Chaple et al., 2015) and **Fuzzy Logic methods** (Chacon-Murguia et al., 2017; Ziółko et al., 2017). Although edge detection methods are effective in defining the boundaries of objects, one limitation across the most literature is the over segmentation problem. This can frequently happen when a number of edges go across two overlapping or merging objects. Another important issue is selecting the right contour representation which can be computationally expensive for large images in terms of processing and memory requirements (Chaple et al., 2015; Singh and Singh, 2015). The following Table 2.1 presents a comparative study of object segmentation methods.

Table 2.1: Comparative study of Object Segmentation Methods

Methods	**Merits**	**Weakness**	**References**
CT	Simple and widely used, low computational cost	Very sensitive to the threshold setting. Difficult to set the exact threshold. Segmented results are often distorted when the ball is against a complex background, under uneven lighting or colour merging with nearby objects.	(Fitriana et al., 2016; Qazi et al., 2015; Bao et al., 2012; Chen et al., 2010, 2011; Wong and Dooley, 2010; Zhang et al., 2010; Arenas et al., 2009)
BS by **Modelling**	Perform well for static background	Computationally expensive due to the need to cope with the expansion	(Wu et al., 2017; Zhao et al., 2017; Yadav et al., 2014;

		of scale, translation, rotation, and deformation. Easily affected by noise factors, such as moving crowds, cluttered background and luminance changes. Camera motion distorts the background model	Tong-yao, 2011; Bayona et al., 2010; Zivkovic, 2004; Toyama et al., 1999)
BS by **Frame Differencing**	Simple implementation, Low complexity, Low memory requirement, Computationally efficient	Provides incomplete object if the object features are not much different between consecutive frames. Poor segmentation when multiple motions or no motion	(Abdelli and Choi, 2017; Srivastav et al., 2017; Tanaka and Miura, 2017; Sengar and Mukhopadhyay, 2016; Yilmaz et al., 2006)
Edge Detection	An effective way to define object's boundaries for further analyse	Over-segmentation and difficult to select the right contour representation when objects are merging. Computation expensive for large images in terms of processing and memory requirements.	(Chacon-Murguia et al., 2017; Ziółko et al., 2017; Chaple et al., 2015; Singh and Singh, 2015; Chen et al., 2012; Ma et al., 2010; Seif et al., 2010; Canny, 1986)

2.2.2 Detection

The foundation for tracking requires optimal detection solutions, and the result of object detection is vital in every tracking application. The effectiveness of the object detection algorithms can be complicated due to: loss of information, noise in images, complex object motion, nonrigid or articulated nature of objects, partial and full object occlusions, complex object shapes, scene illumination changes, and real-time processing requirements. A conventional approach for object detection is to use information in a single frame. However, some object detection methods make use of the temporal information computed from a sequence of frames to reduce the number of false detections (Fan et al., 2016). In object detection, there are three most widely reported methods which are **Feature Detection**, **Motion Detection** and **Supervised Learning**. However, it has been found that some researchers (Crabb et al. 2008; Katalenic et al. 2009; X. Chen et al. 2011; Lee et al. 2012; Q. Huang et al. 2012; Sun and Lam 2013) chose to incorporate all of these methods into their approaches and employ as **Combination Methods**.

2.2.2.1 Feature Detection Methods

Most of the literature attempts to exploit the appearance features of objects such as colour, shape, size, and texture for object detection. Selecting the right features plays a critical role in detection. In general, the most desirable property of a visual feature is its uniqueness so that the objects can be easily distinguished in the feature space (Singh et al., 2013). For high accuracy and real-time object detection, features should be discriminative, robust, and easy to

compute (Lee et al., 2012). Feature selection can be made automatically or manually by the user depending on the application domain. Detection algorithms that have predefined colours or shapes are simple enough to be executed even on low-cost embedded computers in real-time (Katalenic et al., 2009).

Colour-based Detection is a common method used for object detecting and tracking as it is simple to implement and inexpensive to process. A video frame is composed of a set of dots known as pixels and the colour of each pixel is represented by a set of values from a colour space (Brezeale and Cook, 2008). Many colour spaces exist for representing the pixel colours in a frame. Two of the most popular are the **Red-Green-Blue (RGB)** and **Hue-Saturation-Value (HSV)** colour spaces. The original RGB colour may not provide optimal processing as it does not directly represent the influence of light intensity. It is almost always better to use a colour space in which an axis is aligned with brightness, such as HSV (Bradski and Kaehler, 2008a). The distribution of colours in a video frame is often represented using a colour histogram, that is, a count of how many pixels in the frame exist for each possible colour (Brezeale and Cook, 2008). **Colour histograms** are stable object features in the presence of changes in scales and shapes (Swain and Ballard, 1991). These are often used for comparing two frames with the assumption that similar frames will have similar counts even though object motion or camera motion will mean that they do not match on a per-pixel basis. If the target detected object is only a few percent of the size of the frame, applying Colour Histogram together with **Sliding Window (SW)** techniques to every possible window is a time-consuming process (Lee et al., 2012). The crudeness of the colour histogram also means that frames

with similar colour distributions will appear similar regardless of the actual content. Despite its popularity, most colour bands are sensitive to illumination variation. As the object may be perceived to have different colours under different lighting conditions, the performance can be degraded where lighting is uneven. For the case of different parts of an object can be illuminated by various light sources at the same time, Linderoth (Linderoth et al., 2013) proposed a **Colour Classifier** by estimating the illumination spectrum and accounting for its effect on the perceived colour. However, their proposed solution is computationally intensive due to it requiring a large number of training data sets per light source. Then the detector can handle any combination of light sources for a large range of illumination intensities.

Shape-based Detection: While colour is used as a feature, shape-based representations such as object edges are also used as features. Algorithms that detect the boundary of the objects usually use edges as the representative feature. Object boundaries usually generate strong changes in image intensities. Edge detection is used to identify these changes (Singh et al., 2013) and contour representation defines the boundary of an object. The region inside the contour is called the Silhouette of the object. Circle fitting algorithms such as **Hough transform techniques** (Kim and Kim, 2017; Hossein-Khani et al., 2011) are used for detecting the silhouette boundary in edge detection. A useful property of edges is that they are less sensitive to illumination changes compared to colour features. The most popular edge detection approach is the **Canny Edge detector** (Canny, 1986) due to its simplicity and accuracy. However, its performance will

be degraded when the target object is merging with a complex background, and it is difficult to detect the edges.

Texture-based Detection: Like edge features, the texture-based technique also can measure the intensity variation of a surface such as smoothness and regularity. There are various texture descriptors: **Grey-Level Co-occurrence Matrices (GLCM's)** (Haralick et al., 1973) (a 2D histogram which shows the co-occurrences of intensities in a specified direction and distance), **Wavelets** (Mallat, 1989) (orthogonal bank of filters), and **Steerable Pyramids** (Greenspan et al., 1994). While the texture features are less sensitive to illumination changes compared to colour, they require more processing steps to generate the descriptors (Gao et al., 2016; Kolkur and Kalbande, 2016).

Audio-based Detection: Whereas many of the Visual-based approaches use features intended to represent cinematic principles, many of the audio-based features are also chosen to approximate the object perception by sound. Audio features can lead to three layers of understanding (Brezeale and Cook, 2008): low-level acoustics, such as the average frequency for a frame, midlevel sound objects, such as the audio signature of the sound when a ball makes while bouncing, and high-level scene classes, such as background music playing in certain types of video scenes. As audio clips are also typically shorter in length and smaller in file size than video clips, they require fewer computational resources to capture and process than visual features (Huang et al., 2012; Zhang et al., 2006). However, audio-based detection can only detect some specific events such as when a ball bounces on the table or when a player strikes the ball.

2.2.2.2 Motion Detection Methods

While motion is widely used for image segmentation, this technique also applies in object detection. Motion within a video is primarily of two types: movement on the part of the objects being filmed, and movement due to camera actions (still, pan, zoom, and other). Optical flow and **Frame Differencing Methods** are commonly used in motion-based detection and tracking applications, but frame differencing methods have lower computational requirements than the optical flow methods (Brezeale and Cook, 2008).

Optical flow is an estimation of motion in a sequence of images calculated from the velocities of pixel brightness patterns due to object motion or camera motion (Brezeale and Cook, 2008). The optical flow provides accurate object detection over other methods like feature detection and many more. The optical does not provide in motion trajectory instead it gives the information about object direction and movement in the form of vectors (Kale et al., 2015). Applying optical flow to an image gives flow vectors of the points corresponding to the movement (Oklobdzija, 2001). It is a dense field of displacement or motion vectors which defines the translation of each pixel in a region. This could be due to object motion or camera motion. Camera motion is mostly detected by using an optical flow method and is costly to calculate (Brezeale and Cook, 2008). Although optical flow methods are sometimes used for the detection of ball position (Mukai et al., 2011), this method requires iteratively processes such as a brute force search and post-processing (Agarwal et al., 2016).

Frame Differencing means detecting object motion by comparing pixels between consecutive frames. Frame differencing reduces the number of false

detections by enforcing object detection in the regions where the motion occurs. Measuring motion using frame differencing produced results similar to those that measured motion using optical flow, yet frame differencing is more straightforward to implement and less computationally expensive (Abdelli and Choi, 2017; Srivastav et al., 2017; Tanaka and Miura, 2017; Sengar and Mukhopadhyay, 2016).

2.2.2.3 Supervised Learning Methods

Object detection can be performed by learning different object views automatically from a set of examples by means of a supervised learning mechanism such as **Adaptive Boosting** (Zhang et al., 2017), **Decision Trees** (Bu et al., 2009), **Support Vector Machines** (Zheng et al., 2012; Huang et al., 2006), **Gaussian Mixture Model** (Huang et al., 2012) and **Neural Networks** based ball detection (Wong, 2008). Although these methods are popular for their robust classification, learning of different object views waives the requirement of storing a complete set of templates. Given a set of learning examples, supervised learning methods generate a function that maps inputs to desired outputs. However, supervised learning methods usually require a large collection of samples (training sets) from each object class which causes computational overhead. Additionally, this collection requires substantial human effort (physical initialization) in labelling and the necessity of training is a time-consuming process.

2.2.2.4 Combination Methods

While relying on only one feature is not enough for robust object detection due to noise such as colour merging or shape distortion, detection procedures can be improved by incorporating a variety of features. Although colour, shape, and area are commonly employed in the object detection process (Chen et al., 2011), some researchers (Sun and Lam, 2013; Huang et al., 2012; Lee et al., 2012; Chen et al., 2011; Katalenic et al., 2009; Crabb et al., 2008) chose to incorporate a variety of visual, audio and text features into their approaches. According to their experimental results, they can improve the detecting performance and overcome the weaknesses of each. Complementary to this, Table 2.2 presents a comparative study of object detection methods and their techniques which is obtained by based on their type of methods, and requirement of training and manual initialization.

Table 2.2: Comparative study of Object Detection Methods

Methods	**Merits**	**Weakness**	**References**
Feature Detection	**Colour:** Simple, Obvious feature, widely used	**Colour:** Suffer when images are under different illumination conditions. **Colour histograms:** not suitable for a small object. **Sliding Window (SW) Technique***:* Computational complexity**.** **Colour Classifier:** Time-consuming	(Fitriana et al., 2016; Linderoth et al., 2013; Lee et al., 2012; Hsin et al., 2011; Katalenic et al., 2009; Brezeale and Cook, 2008; Swain and Ballard, 1991)

	Shape: Less sensitive to illuminating changes	**Shape:** its performance will degrade when the target object is merging with a complex background and difficult to detect the edges	(Kim and Kim, 2017; Xiao and Yilmaz, 2016; Cui et al., 2010, 2013; Hossein-Khani et al., 2011; Ishii et al., 2011; Leo et al., 2008; Canny, 1986)
	Texture: Less sensitive to illuminating changes	**Texture:** it requires more processing step to generate the descriptors	(Gao et al., 2016; Kolkur and Kalbande, 2016; Sheu et al., 2010; Greenspan et al., 1994; Mallat, 1989; Haralick et al., 1973)
	Audio: it requires fewer computational resources to obtain and process than visual features.	**Audio:** can detect only some specific events	(Huang et al., 2012; Brezeale and Cook, 2008; Zhang et al., 2006)
Motion Detection	**Optical flow:** widely used to detect the object motion	Costly to calculate, requires iteratively process such as brute force search and post-processing	(Agarwal et al., 2016; Mukai et al., 2011; Brezeale and Cook, 2008; Oklobdzija, 2001)

	Frame Differencing: does not require predefined pattern templates, Low complexity	Struggles to identify a non-moving object, degrade the performance when multiple motions or no motion	(Abdelli and Choi, 2017; Srivastav et al., 2017; Tanaka and Miura, 2017; Sengar and Mukhopadhyay, 2016; Yilmaz et al., 2006)
Supervised Learning	Powerful and Robust methods E.g., **Neural Networks**, **Adaptive Boosting**, **Decision Trees**, **Support Vector Machines** and **Bayesian networks, Gaussian Mixture Model**.	High memory requirement, computationally expensive, requires substantial human effort such as manual labelling.	(Zhang et al., 2017; Zheng et al., 2012; Bu et al., 2009; Wong, 2008; Huang et al., 2006)
Combination Methods	Improve detection performance, to overcome the weaknesses of each method	Need a careful selection of essential features as involving too many features will complex the method (**Overfitting**)	(Sun and Lam, 2013; Huang et al., 2012; Lee et al., 2012; Chen et al., 2011; Katalenic et al., 2009; Crabb et al., 2008)

2.2.3 Tracking

Object tracking is the next step following object detection. Object locations in every frame are obtained by means of object detection algorithms, and then the tracker predicts the location of an object over time by using regional information obtained from previous frames. In short, tracking can be defined as estimating the trajectory of an object in every frame of the video. Although numerous approaches have been proposed in object tracking, they can be divided into three broad categories: methods using primitive geometric models (**Kernel-based Tracking**), methods using contour evolution (**Silhouette-based Tracking**) and methods establishing point correspondence (**Point-based Tracking**). These algorithms differ regarding the appearance representation (single view or multi-view) used, the number of objects tracked, and the method used to estimate the object motion.

2.2.3.1 Kernel Tracking

Kernel tracking is typically performed by computing the motion of the object, which is represented by a primitive object region, from one frame to the next. **Mean Shift Tracker** (Comaniciu and Meer, 2002), **Support Vector Machine** (Zheng et al., 2012; Huang et al., 2006), and **Layer based tracking** (Zhou et al., 2013, 2015) are parts of Kernel Tracking. These techniques involve representation of an object, object features, shape, and appearance of an object. For example, the kernel can be a rectangular template or an elliptical shape. However, the objects may appear different from different views, and if the object view changes dramatically during tracking, the appearance model may no longer

be valid, and the object track might be lost. For example, the popular mean shift tracking algorithm assumes that the target object has to separate sufficiently from the background, but this assumption is not always true, especially when tracking is carried out in dynamic environments such as in sport videos (Teachabarikiti et al., 2010). Moreover, Kernel tracking algorithms are iterative methods and their computational overhead and time consumption are not suitable for real-time tracking.

2.2.3.2 Silhouettes Tracking

Silhouette tracking is widely used for detecting and tracking complex nonrigid shapes (Xiao and Yilmaz, 2016; Leo et al., 2008). It is performed by estimating the object region in each frame with the help of an object model obtained by the previous frames. It uses the information encoded inside the object region used to represent the model. Silhouette tracking is usually carried out by BS. Once the object silhouettes are extracted, matching is performed by some distance measurement computing techniques such as **Euclidean Distance** (Dokmanic et al., 2015) between the object models associated with each silhouette. Object models are usually in the form of **Density Functions** (colour or edge histograms), **Silhouette Boundary** (closed or open object contour), **Object Edges** or a combination of these models. This means some algorithms only use information about the silhouette boundary for tracking, while others use the entire region inside the silhouette. The advantage of silhouette tracking is their flexibility to handle a large variety of object shapes. For the objects whose shapes

can be stable as rectangles or ellipses in every frame, **Contour Matching** and **Shape Matching** methods are widely used for Silhouette Tracking.

2.2.3.3 Point-based Tracking

The main difference between silhouette tracking and point based tracking is the way the object representations and the models are used. In particular, silhouette tracking uses the complete object region while point based tracking uses points. In addition, silhouette matching makes use of an object's appearance features, whereas point matching uses only position-based features. In Point based tracking, tracking can be formulated as the correspondence of detected objects represented by points across consecutive frames. For tracking objects, which appear very small in an image, point representation is usually appropriate.

Motion models have been widely used in points based tracking (Maksai et al., 2016; Seo and Wuest, 2016; Ratnayake and Amer, 2015; Takahashi et al., 2015) because of their relative simplicity and low computational cost. It is a technique used to track and detect the travelling direction of objects. It conducts the tracking by taking a statistical measurement that uses the state space approach to model the object properties such as position, velocity, and acceleration. The mathematical equation of object trajectories is normally assumed as straight lines for short periods and its position is predicted by using velocity vectors of previous images. Measurements usually consist of the object position in the image, which is obtained by detection mechanisms.

Kalman filter (KF) and **Particle filter (PF)** are commonly used in point based tracking algorithms (Pan and Niemeyer, 2017; Takahashi et al., 2016;

Wang et al., 2016; Kim and Kim, 2009; Pinho et al., 2006). A Kalman filter which is also known as **linear quadratic estimation (LQE)** is used to estimate the state of a linear system where the state is assumed to be distributed by a **Gaussian noise** (R. E. Kalman, 2001). Kalman filtering is composed of two steps, prediction, and correction. The prediction step uses the **State Model** to predict the new state of the object's position and speed. When the case of the detected object is non-linear, **Extended KF** (Yilmaz et al., 2006) plays the role of detection and tracking. One limitation of the KF is the assumption that the state variables are normally distributed (Gaussian). Thus, the KF will give poor estimations of state variables that do not follow a Gaussian distribution (Yilmaz et al., 2006). However, its implementation with other features and classifiers can make a better solution (Anuj and Krishna, 2017).

To summarise, the tracker using KF is useful in detecting moving object in similar background, but it is not able to track low-resolution objects, and it cannot track objects with variation in speed of movement (Kale et al., 2015). **Kernel base tracking** and **Silhouette-based tracking** require detection only when the object first appears on the screen whereas point-based tracking involves detection in every frame. Table 2.3 presents a qualitative comparison of tracking methods.

Table 2.3: Comparative study of Object Tracking Methods

Methods	**Merits**	**Weakness**	**References**
Kernel Tracking	Powerful and Robust methods such as **Mean Shift Tracker**, **Support Vector Machine** and **Layer based tracking** are parts of Kernel Tracking.	Iterative methods, its computational overhead and time consumption are not suitable for real-time tracking.	(Zhou et al., 2013, 2015; Zheng et al., 2012; Teachabarikiti et al., 2010; Huang et al., 2006; Comaniciu and Meer, 2002)
Silhouette Tracking	Widely used for complex nonrigid shapes, flexibility to handle a large variety of object shapes	Suffer when images are merging with complicated background and cannot extract their shape / contour.	(Kim and Kim, 2017; Xiao and Yilmaz, 2016; Hossein-Khani et al., 2011; Leo et al., 2008)
Point-based Tracking	Simplicity and low computational cost. For tracking objects, which appear very small in an image, point representation is usually appropriate.	Poor estimation and require reinitialization techniques when the object do not follow its distribution or has abrupt directional change.	(Pan and Niemeyer, 2017; Seo and Wuest, 2016; Wang et al., 2016; Ratnayake and Amer, 2015; Takahashi et al., 2015; Kim and Kim, 2009; Pinho et al., 2006)

2.3 Overview of technologies applied in various sports

Technologies have been applied in many sports events for decades (Wong, 2008) as the development of high-speed digital cameras and video processing has attracted people's attention in sports video analysis. Even though computer vision based analysis of sport videos has been addressed by many authors (Maddalena & Petrosino, 2013); Tamaki et al., 2012; Chen et al., 2011; McIlroy, 2008; Owens et al., 2003), it is still a hot topic within the multimedia video analysis community (Hawk-Eye Innovations, 2018; BBC, 2018; Kim and Kim, 2017; Shahjalal et al., 2017; Arenas et al., 2009) as sport is an evergreen field and attracts big spending each year (Shih, 2017). This section reviews current computer vision technologies applications in various sports video analyses and discusses the research issues of the field and the potential applications. Examples include computer-assisted referee such as Goal-line and Hawk-eye systems (Hawk-Eye, 2018; Tsang, 2013; Bal and Dureja, 2012; Owens et al., 2003).

2.3.1 Cricket

In Cricket, a vision based technology, called **A-Eye**, which automates the role of the third umpire has been proposed and implemented in (Mahmood et al., 2011). The **Graphical User Interface (GUI)** of A-Eye is as shown below in figure 2.1.

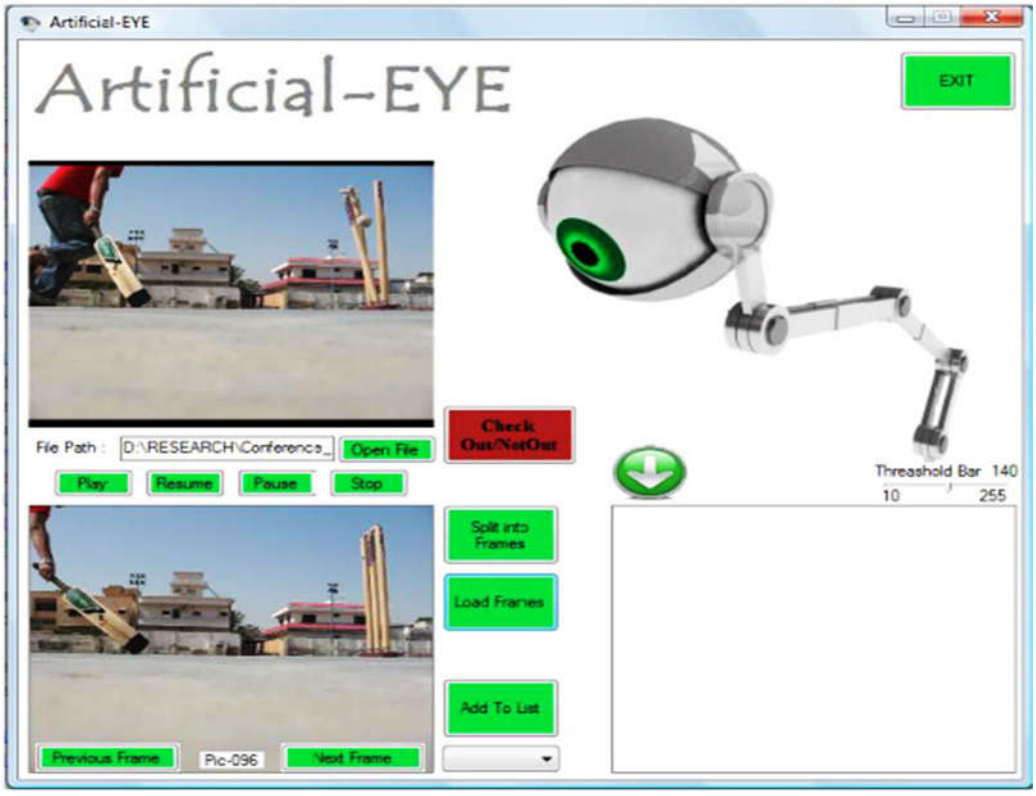


Figure 2.1: Artificial-Eye GUI (Mahmood et al., 2011)

A-Eye technology can be utilised during play by installing a (hidden) camera that is located on the surface of the ground (zero height) and facing the wicket. A-Eye was implemented as a desktop application in the **C-Sharp (#)** programming language. It was adopted as a **three-tier approach**, i.e., the client tier, the middle tier and the back-end tier. The architecture of A-Eye will be described in figure (2.2) as below.

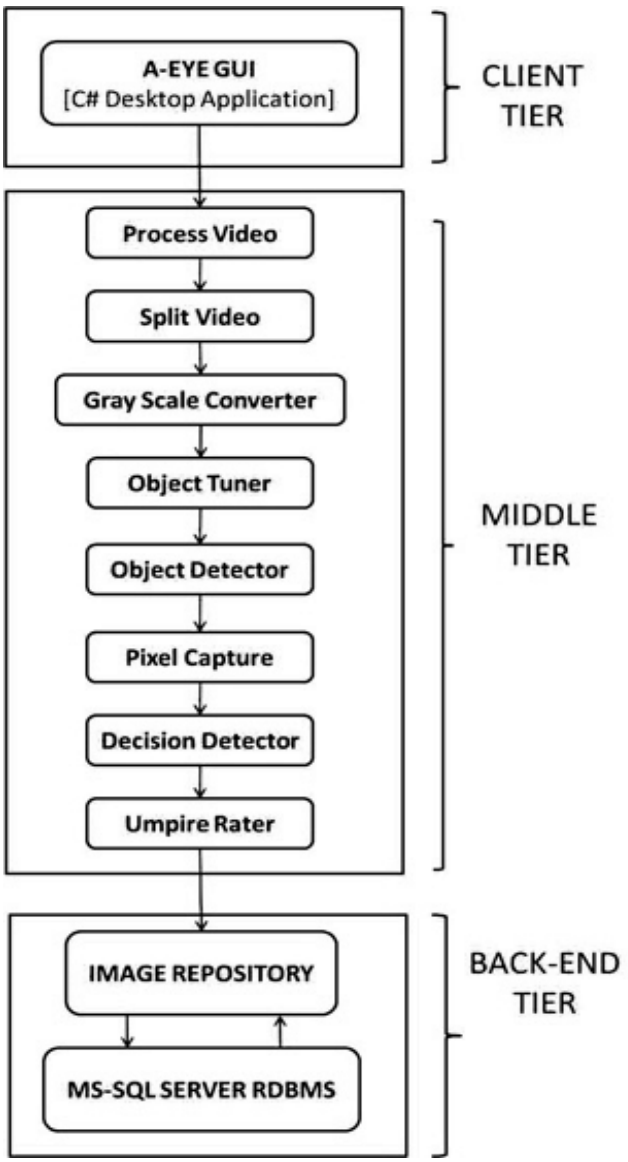


Figure 2.2: Architecture of A-Eye (Mahmood et al., 2011)

To effectively apply image processing techniques, there are two types of image conversion. The first is to convert the incoming frame into a binary format, i.e., into black and white, and the second is to transform into a discrete number of shades of grey, i.e., into the greyscale mode. In A-Eye, the input video frames were firstly converted to grey scale because a greyscale image supports more accurate detection of objects as compared to a black-and-white image (Gonzalez

and Woods, 2002). Then **Hough Transform** was used for detecting the crease marker. The **Motion Detection Algorithm (MDA)** which was based on a simple comparison of the pixels across consecutive frames was used for detecting four objects in motion, i.e., the player, the bat, the ball, and the wicket. Once it identified the motion regions in a frame, a technique known as blob was used for counting to determine the number of detected objects. A set of 30 run-out personally filmed videos in 2D was used for experimentation. The Umpire decision module of A-Eye was used for calculating a rating for the performance of the field umpires. The performance of A-Eye was validated by comparing it with the performance of the third umpire. The results have been proved that it has a potential to minimize decision errors made by third umpires and was able to estimate a rating for the field umpires.

2.3.2 Football

A robot referee intended to be used in the **RoboCup** humanoid league for robot soccer was proposed in Arenas's work (Arenas et al., 2009). Their idea is implemented using a service robot that moves along one of the field sides, uses its own cameras to analyse the game, and communicates its decisions to the human spectators using speech, and to the robot players using wireless communication. The robot uses a video-based game analysis toolbox that is able to analyse the actions at up to 20 fps. This toolbox includes robots, ball, landmarks, and lines detection and tracking, as well as refereeing decision making.

In their detection system, all the objects of interest for the soccer game (field carpet, field and goal lines, goals, beacons, robot players, ball) are detected using **Colour Segmentation** and some simple **Rules**. To detect the robots at different scales, a **Multi-Resolution Analysis** of the images is performed, by downscaling the input image by a fixed scaling factor. Windows of 24 x 24 pixels are extracted in the **Window Extraction module** for each of the scaled versions of the input image. The windows are analysed by a **Nested Cascade of Boosted Classifier (Cascade Classification Module)**. The block diagram of the detection system is shown below in Figure 2.3.

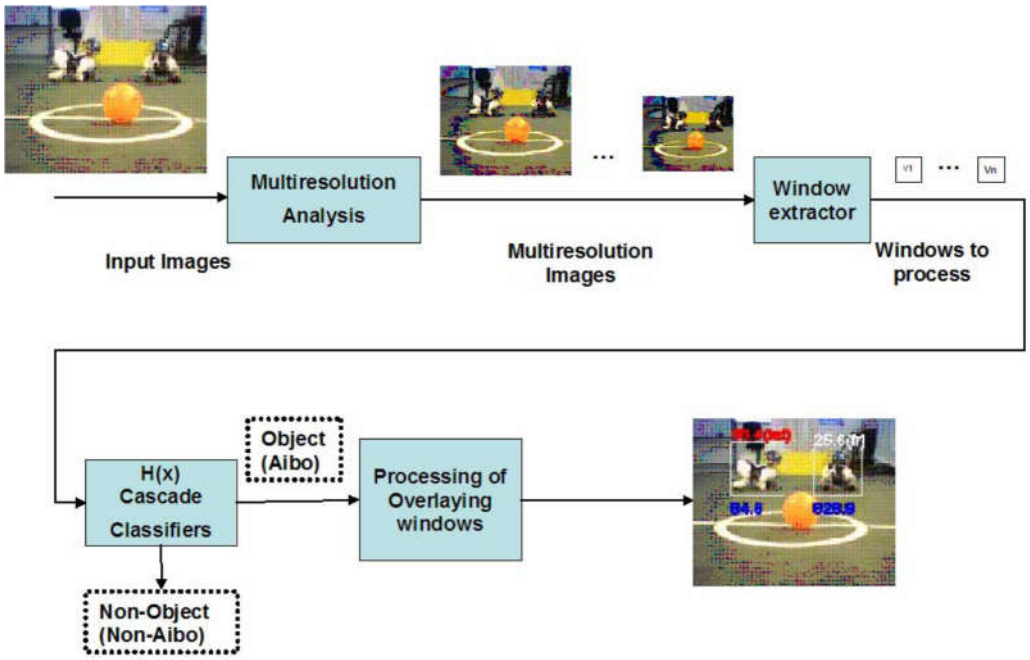


Figure 2.3: Block diagram of the detection system (Arenas et al., 2009)

The system is composed by seven main modules Object Perception, Visual Tracking, Self-localization, Refereeing, Motion Control, Speech Synthesis, and Wireless Communications, and makes use of two databases: Rules

(input) and Game Statistics (output). The block diagram of the robot referee controller is shown in figure 2.4.

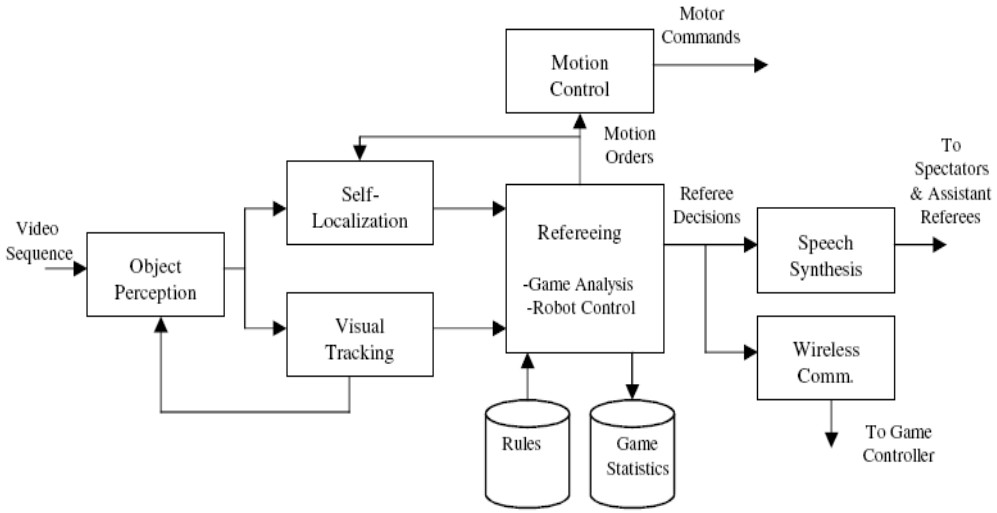


Figure 2.4: Block diagram of the robot referee controller (Arenas et al., 2009)

The Visual Tracking module is in charge of tracking the moving objects, i.e. the ball and the robot players. The implemented ball and player tracking system is built using the **Mean Shift Algorithm**, applied over the original image. The KF is employed to maintain an actualised feature model for mean shift. The Refereeing module is in charge of analysing the game dynamics and the actions performed by the players (e.g. kicking or passing), and detecting game relevant events (goal, ball out of the field, illegal defender, etc.). This analysis is carried out using information about static and moving detected objects, and the game rules, which are retrieved from the Rules database. Refereeing decisions (e.g. goal was scored by team A) are sent to the Speech Synthesis and Wireless Communication modules, motion orders that are sent to the Motion Control module, and game statistics (e.g. player 2 from team A score a goal) that are

stored in the corresponding database. This robot system is validated and characterized in real game situations with humanoid robot players. In their future implementations, they are planning to use multiple cameras to have more information of the activities in the field. Although their system is claimed to able to track all game moving objects and the lines in near real-time (up to 20 fps), satisfactory acquisition of the ball trajectory requires a high frame rate (at least 50 frames per second) to reduce the problems caused by camera calibration and to reduce blur (Huang et al., 2012). The lower frame rate created a larger timing gap between frames, which is manifest as changes of shape and size of the image of the ball, resulting in a high number of false candidates and hence poor tracking performance.

2.3.3 Tennis

One of the early approaches of trajectory-based ball detection and tracking algorithm for broadcasting tennis video can be found in Yu's paper (Yu et al., 2004). In that approach, the identified non-ball objects were firstly removed by filtering size, straight line, colour, shape, circularity, and isolation. After that, candidate classification method was used for detecting candidate objects for each frame. Then, the **KF-based procedure** was used for producing the candidate trajectories from each candidate feature image. After that, each candidate trajectory was evaluated by identifying the ball trajectories through a selection procedure. Unlike the other object-based algorithm, Yu's approach (Yu et al., 2004) did not decide whether an object is a ball. Instead, they decided whether a candidate trajectory was a ball trajectory. In Mukai's work (Mukai et al., 2011),

tennis plays are analysed, and skills of players are evaluated quantitatively by computer vision technology. Two high-definition TV cameras (right and left) are used to detect 3D ball position and 2D player position. The ball trajectories and player positions of tennis singles games are detected from Stereo Images, and six skill parameters are calculated. The skill factors, as listed in Table 2.4, were analysed by regression of the skill scores from human evaluation with the player skill parameters which present in figure 2.5.

Table 2.4: Six player skill evaluation method (Mukai et al., 2011)

1) Distance between ball bounce point P_i and player: ℓ
2) Distance between ball bounce point P_i and side line: d
3) Net clearance: c
4) Distance between present ball bounce point P_i and previous ball bounce point P_{i-1}: δp
5) Ball shot speed: v
6) Ball shot height of service: h

The parameters have following meanings.

1) ℓ: If this value is large, the player must move a great distance to hit the ball. Therefore, the larger this parameter is, the higher the skill score.
2) d: It is difficult for players to return the ball that bounces near the sideline. Therefore, the smaller this parameter is, the higher the skill score.
3) c: It is difficult to return the ball that comes over the net with a small clearance. Therefore, the smaller this parameter is, the higher the skill score.
4) δp: If this value is large, the receiver must run in a different direction to hit the ball. The larger this parameter is, the higher the skill score.

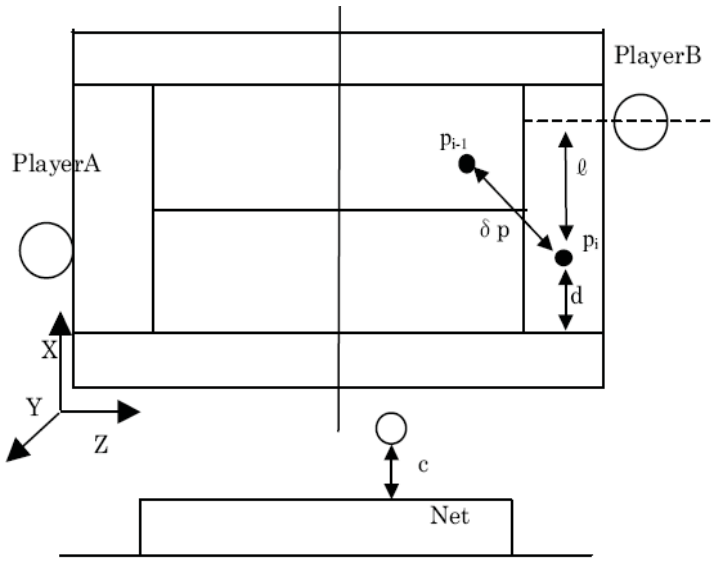


Figure 2.5: The player skill parameters (Mukai et al., 2011)

To eliminate misdetections caused by ball shadows on the tennis court, the ball position in the right and left images is detected by using three successive **Images Subtraction Method** which is known as **Frame Differencing** (Mukai et al., 2011). The three images are previous, current and following frame images. Sequential images of 64 return strokes and 15 services by four players were analysed quantitatively. Since this was a pilot research, they only evaluated return strokes and services rather than building a complete system. In reality, it is necessary to evaluate all game elements including volleys and smash shots and those will be included in the further study of their research.

2.3.4 Table Tennis

Since this research is to investigate the best approach that can be effectively used for detecting and tracking a fast-moving table tennis ball for developing an automatic umpiring system, this section presents a wide coverage of the state-of-

the-art research work regarding the technological development in table tennis. In fact, developing a table tennis match umpiring system is a very challenging task because it needs to produce the judgment within a few seconds or sooner after the rally is complete. The umpiring process involves a series of fast actions and is strictly governed by the Laws of Table Tennis stated in the ITTF Handbook (ITTF, 2018). It is even very difficult for human umpire since it often requires lots of professional judgment. The demand for timely and accurate observations of rallies imposed by the table tennis rules is also very high. A typical example of judging difficulty is to determine whether the ball hits the edge of the playing surface, which is a legal, or the side of the table, which is a foul. The prior development of high-motion table tennis ball tracking for umpiring applications can be found in Wong and Dooley's papers (Wong and Dooley, 2010, 2011; Wong, 2007, 2008, 2009). Their research was aimed at assisting table tennis umpires to make an accurate judgment about services. Their pilot study mainly covered one (2.06.02) of many table tennis rules. The remaining rules were not considered and hence umpiring a table tennis match is still a fertile research area. The rules shown below in table 2.5 are directly related to services.

Table 2.5: Table- tennis rules regarding the service

Index	Description
2.06.01	Service shall start with the ball resting freely on the open palm of the server's stationary free hand.
2.06.02	The server shall then project the ball near vertically upwards, without imparting spin, so that it rises at least 16 cm after leaving

	the palm of the free hand and then falls without touching anything before being struck.
2.06.03	As the ball is falling the server shall strike it so that it touches first his court and then, after passing over or around the net assembly, touches directly the receiver's court; in doubles, the ball shall touch successively the right half court of server and receiver.
2.06.04	From the start of service until it is struck, the ball shall be above the level of the playing surface and behind the server's end line, and it shall not be hidden from the receiver by the server or his doubles partner or by anything they wear or carry.
2.06.05	As soon as the ball has been projected, the server’s free arm shall be removed from the space between the ball and the net. Note: The space between the ball and the net is defined by the ball, the net and its indefinite upward extension.
2.06.06	It is the responsibility of the player to serve so that the umpire or the assistant umpire can see that he complies with the requirements for a good service.
2.06.06.01	If the umpire is doubtful of the legality of a service he may, on the first occasion in a match, declare a let and warn the server.
2.06.06.02	Any subsequent service of the doubtful legality of that player or his doubles partner will result in a point to the receiver.
2.06.06.03	Whenever there is a clear failure to comply with the requirements for a good service, no warning shall be given, and the receiver shall score a point.

From 2007 to 2011, Wong and Dooley (Wong and Dooley, 2010, 2011; Wong, 2007, 2008, 2009) tried to find the possible solutions for tracking the table tennis ball from live video images by experimenting various filtering, image segmentation and enhancement techniques. In the earlier stage of Wong's research (Wong, 2007), **Artificial Neural Network (ANN)** is used for classifying the ball among the similar (color, size and shape) objects and identify whether the detected object is a ball on the palm, a ball in mid-air, or not a table tennis ball. While satisfactory results were achieved, Wong only covered the service part of the rallies, in which the ball is not travelling at very high speed. Moreover, ANN consumes a lot of processing time in training and requires substantial human effort such as manual labelling.

In 2008 (Wong, 2008), Wong adopted a frame-based object segmentation method called **Two-Pass Threshold Method (TPT).** TPT uses different thresholds in each pass with the first applying a coarse threshold followed by a relaxed threshold. In this method, a high threshold is firstly applied for removing most irrelevant objects and a low threshold is secondly applied for fully revealing the boundary of the objects detected in first pass. The benefit of TPT is it loosens threshold selection so the value in each pass can be less precisely set. To assess which candidate ball is the **Object-Of-Interest (OOI),** his system comprised a suite of algorithms that adaptively exploit spatial and temporal information such as size (maximum width, maximum height), shape (roundness, rounded upper contour), area (perimeter), motion and trajectory. His point-based and neural networks approaches were based on an image taken from a single angle which could not tackle the occlusion problem.

Therefore, Wong suggested in his 2009 paper (Wong, 2009) that the problem would be mitigated if multiple cameras are employed to get views from different angles. However, larger amounts of data feeding from multiple cameras will increase pressure on the processing workload and time. To improve time performance, he suggested implementing the system as a rule-based multi-agent system (Wong and Dooley, 2010, 2011) with artificial intelligence techniques. In this way, several tasks can be executed in parallel, for example, one agent can identify the ball while other can be checking the location of the ball and another can do the measurement. By working together, the agent system can simultaneously make several observations.

For humanoid robots to play table tennis, a **Physical Bouncing Model** for predicting a trajectory of the table tennis ball (ping-pong) has been proposed by Bao (Bao et al., 2012). Four high-speed cameras and two low-speed cameras were used to grab and transfer images to a powerful vision processing industrial personal computer (PC). The sample rate of the cameras can reach up to 200 (fps), and a signal generator is used to synchronize these cameras as shown in figure 2.6.

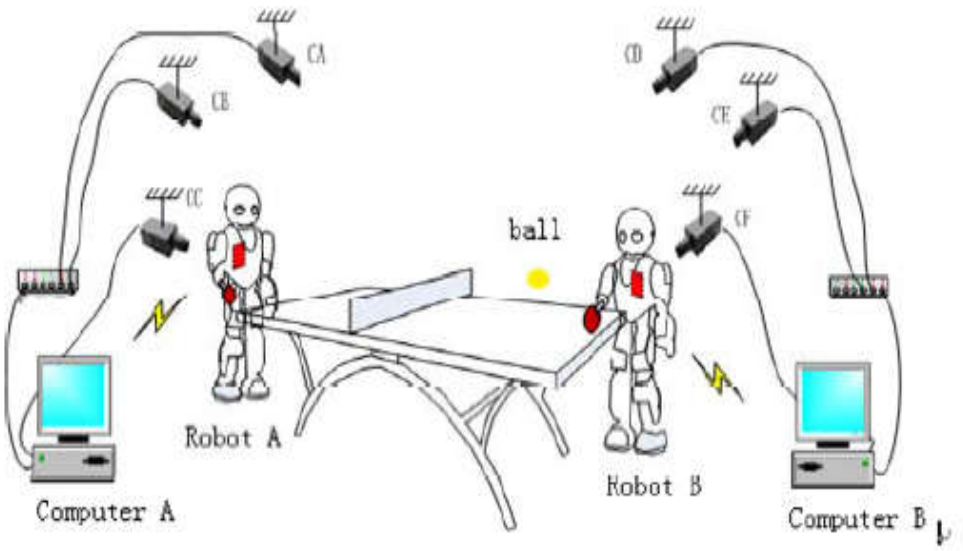


Figure 2.6: Multiple Camera-based Vision System (Bao et al., 2012)

In their system, each camera pair covers one half of the ping-pong table and points at the opposite side so that the ball can't be occluded by player or robot. The whole ping-pong table can be observed and then the whole ping-pong trajectory can be detected by merging the data from the two stereo pairs. Image acquisition, image processing, reconstruction and trajectory prediction are done on the computer. A **nonlinear bouncing model** based on the conservation of **Momentum Theorem and Moment of Momentum Theorem** to include the spin of the ball describes the collision between the ball and the table. The candidate hitting position and time are sent to the robots on both sides by the wireless communications module.

The table tennis ball is governed by **Newton's Laws of Motion**. Many players create a heavy spin on the ball and it has been observed that certain world-class players had imparted spins on ping pong balls that were around 9,000 revolutions per minute (rpm) (Table Tennis Master, 2017). A skilful table tennis

player can spin the ball exceeding 5000 rpm, while it is typically around 3000 rpm for a novice (Tamaki et al., 2012). A method for estimating the rotational velocity from the real image sequences of table tennis rallies has been developed by (Tamaki et al., 2012). Their spin measurement method is based on **Inverse Compositional Image Alignment (ICIA)** which accelerates computation. Wang (Wang et al., 2012) proposed a model of human behaviour imitation for a robot to play table tennis. Their main strategy is to record a video of the action in which people played table tennis, then to analyse the video of the bat trajectory. The movement of the robot's racket is fitted through people hitting ball according to the trajectory of the racket. Since the key of the whole work is mainly about the racket, the first step of their research is extracting the racket in each frame of video by thresholding the red part of the image as shown in figure 2.7.

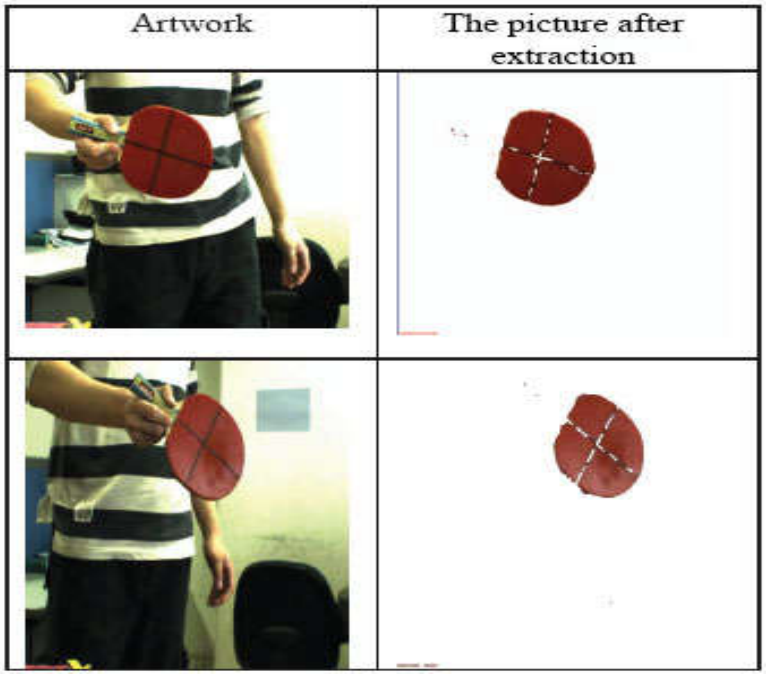


Figure 2.7: Extraction of a racket (Wang et al., 2012)

It can be found that the RGB was heavily influenced by light. Therefore, Wang (Wang et al., 2012) extract the racket in image using HSV instead of RGB. After smoothing the image to eliminate distortion points, a **Canny Edge Detector** is used for detecting the range of edges in the grey-scale image as shown in figure 2.8.

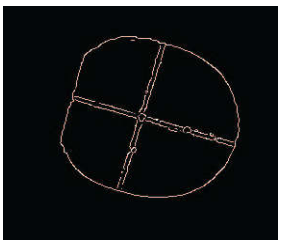


Figure 2.8 (a): Image after edge detection

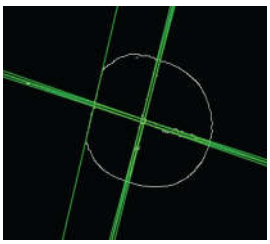


Figure 2.8 (b): Hough transform to obtain a straight line

Figure 2.8: Edges Detection (Wang et al., 2012)

After that, the **Hough transform** is used for detecting the straight lines in the edge image. Then 3D coordinates of the centre of the racket and racket posture are obtained via the **PNP (Perspective-N-Point)** positioning approach based on the intrinsic parameters of the camera. For calculating the centre and the posture of the racket, 3D coordinates of points are fitted by the **RANSAC (Random Sample Consensus)** algorithm.

Another interesting approach of predicting the trajectory of the table tennis ball and its placement by using computer simulation for wheelchair player can be found in Chiu's work (Chiu, Ching-Hua, et al., 2010). Because wheelchair table tennis players are physically limited by their disability, they have to be trained efficiently for acquiring excellent skills and remarkable performance.

Two integral skills they must possess are the control of ball placement and tactics of competition.

Figure 2.9: Wheelchair table tennis singles (Chiu, Ching-Hua, et al., 2010)

To help such players, their study aimed to design a system which would be capable of predicting ball placement in table tennis singles. To this end, they adopted the **Back-Propagation Neural Network (BPNN)**, whose structure consisted of an input layer (48 input neurons), a hidden layer (30 hidden neurons), and an output layer (12 output neurons). The ball placement parameters were first converted into training samples of the BPNN and then the learning algorithm of the BPNN was subsequently applied to the training samples. Finally, a recalling algorithm was used for predicting ball placement. (Yingzhu Li et al., 2010) proposed a real-time immersive table tennis game for two players with motion tracking. Additionally, a **physics-based ball animation model** is designed for the game, which includes fast detection of the ball colliding with table, net and

quick moving rackets. To achieve high-speed movement tracking of rackets and player viewpoints, hybrid inertia and ultrasonic sensing technology are used, with each player holding a hand tracker as grasping a racket and wearing a head tracker. To achieve visual immersion, two large rear projection stereoscopic screens as shown below in Figure 2.10 are used to provide an individual view for each player based on the player's perspective.

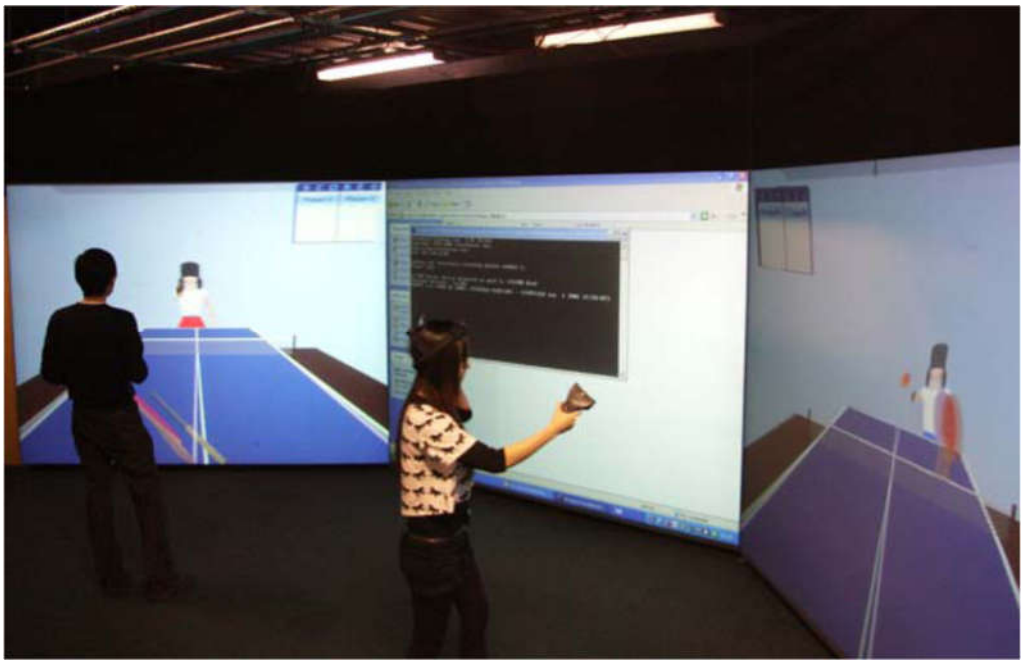


Figure 2.10: Two Players Play against Each Other (Yingzhu Li et al., 2010)

By wearing a pair of polarised glasses, each player is able to see his/her own virtual racket, a standard table tennis table, and a flying ball, as well as the opponent's avatar holding a racket, in 3D with an impression of depth. The system is shown to offer some unique features and form a good platform for the development of other immersive games for multiple players. The computing system consists of three PCs as shown below in figure 2.11 and each one runs on an Intel Xeon 3.06 GHz CPU with 2G RAMs and a 256 MB NVIDIA Quadro FX

3000 Graphic Card. One PC is used as the server to run the application program, which is responsible for the InterSense (Wormell and Foxlin, 2003) tracking control, data processing and animation computation. The other two PCs are used as clients, where each one renders the scene according to the computation results sent from the server and drives a pair of projectors to provide an individual stereoscopic display according to the viewpoint of each player.

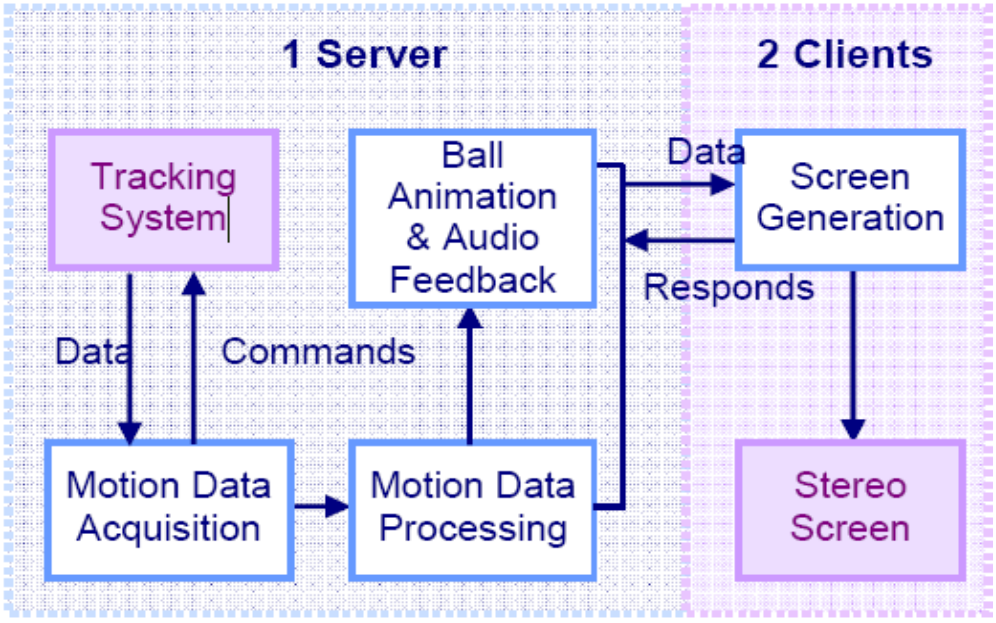


Figure 2.11: Software Modules and Data Flow (Yingzhu Li et al., 2010)

The communication between the server and two clients is based on the TCP/IP protocol, and data is transmitted through a 1G Ethernet connection. For the server computer, it runs the **Motion Data Acquisition Module** to acquire position and orientation data of the head and hand of each player from the tracking system based on the InterSense. The Motion Data Processing Module provides player viewpoints as well as positions and orientations of the virtual rackets and avatars

to be drawn by the two client computers. The Ball Animation and Audio Feedback Module provides the motion of the ball according to simplified physical laws and produces a sound if a collision is detected. For the two client computers, all fixed static virtual objects (e.g. table, wall and floor) are pre-computed, and each one runs its own Scene Generation Module to produce a stereo pair for each screen upon receiving the dynamic object data from the server.

As shown in figure 2.12 and 2.13 a high-speed stereo vision system with two smart cameras which adopt a distributed parallel processing architecture based on a local area network is presented in (Zhang et al., 2010).

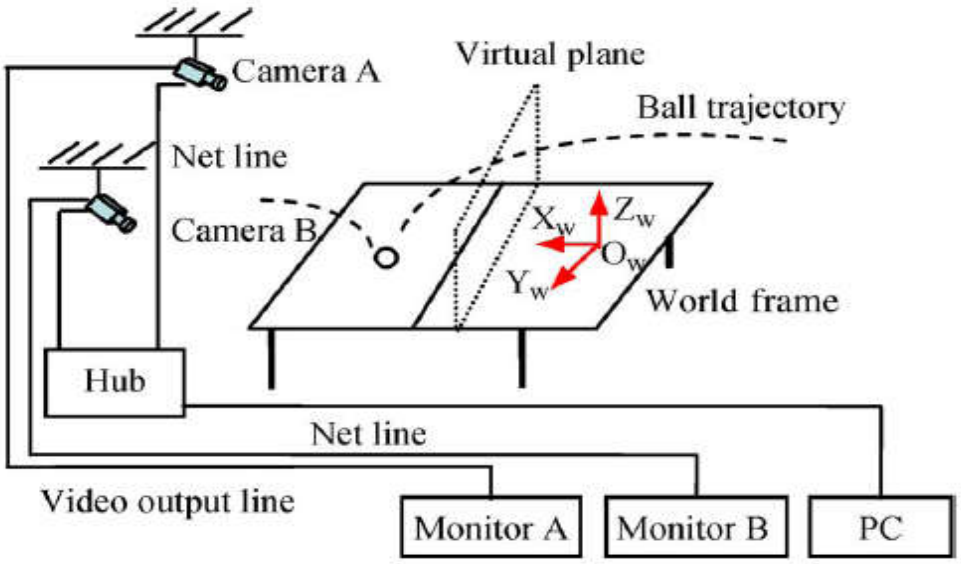


Figure 2.12: Distributed high-speed vision system (Zhang et al., 2010)

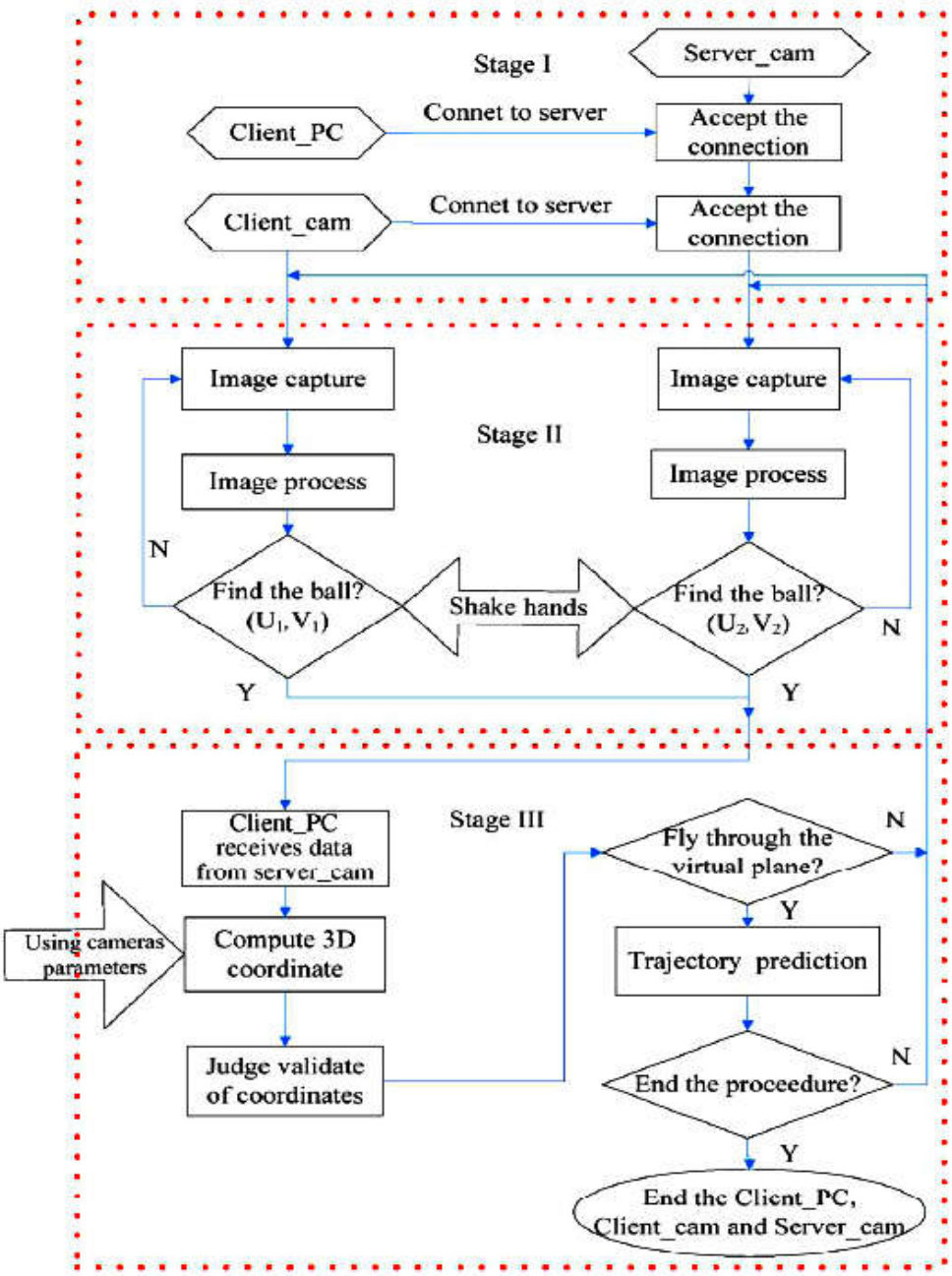


Figure 2.13: Distributed parallel processing architecture (Zhang et al., 2010)

A computer receives the image coordinates of the ball from the cameras via the local area network and computes its 3D positions in the working frame. Then, the flying trajectory of the ball is estimated and predicted according to the measured positions, **Flying and Rebound models**. The main Motion Parameters

of the ball such as the Landing Point and Striking Point can be calculated from its Predicted Trajectory. In order to predict the 3D trajectory of the table tennis ball, the **Aerodynamic Model** of the ball flying in the sky and the **Bouncing Model** are established in Chen's research (Chen et al., 2011). In their system, the **Aerodynamic Model** was used for estimating the trajectories of the flying table tennis ball and **Bouncing Model** was used for getting the velocity changes after bouncing. In the Bouncing Model, the vertical speed and the horizontal speed were treated separately. However, their prediction of the ball trajectory is done by solving only the **Linear equation**. There is no consideration for the **Non-linear equation** and the spin of the ball is neglected.

In the table tennis games, over a hundred points may be included, demanding a further analysis to compare the measurement of each important point. Most of the audiences are concerned not only by the final score but also by the match's highlights and excitement. To provide valuable information for the semantic understanding of sports games, research investigating the ball hit detection in table tennis games is presented in (Zhang et al., 2006) based on audio analysis by employing **Energy Peak Detection (EPD)** and **Mel Frequency Cepstral Coefficient-based (MFCC-based) Refinement (MBR)**.

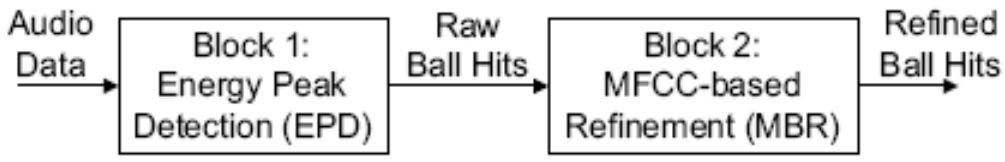


Figure 2.14: Framework of a ball hit detection system (Zhang et al., 2006)

Another system of extracting high-lever semantic features can be found in Chen and Zhang's paper (Chen and Zhang, 2006). Their system was intended for automatically ranking the highlight levers and replaying some important information to referee for scoring a reasonable mark such ball position, table position, the player action, and the ball trajectory.

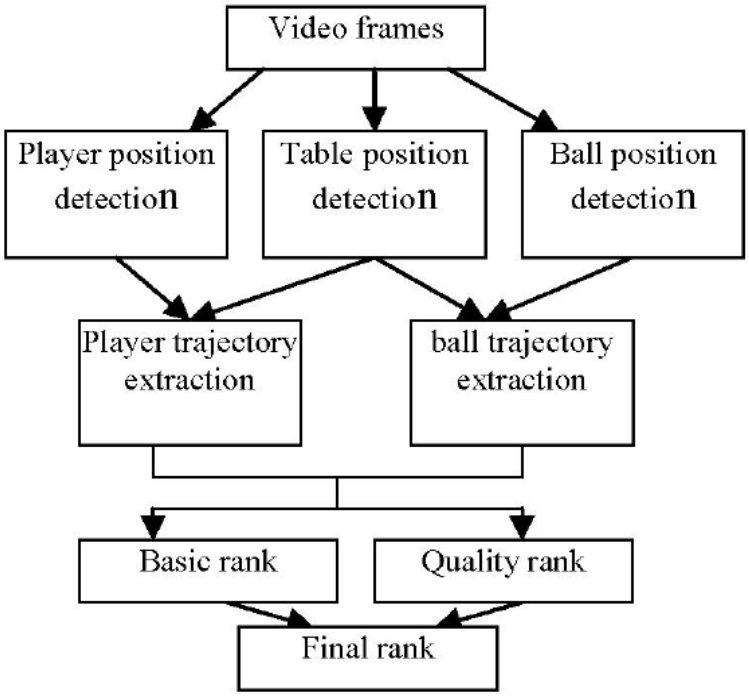


Figure 2.15: Architecture of the framework (Chen and Zhang, 2006)

To derive the ball trajectory in their system, all ball candidates were first detected using a feature-based approach such as colour, shape, size and position limits. The differences between the current frame and the next consecutive frame were calculated first to find out the changing part in the current video. Then a morphological filtering procedure was called to eliminate noise and connect near regions that belong to one object. Then both the motion and appearance

information were employed to find the best ball candidate by a Bayesian decision framework. To derive the ball trajectory, the ball's dynamic and appearance parameters were updated by a KF and an incremental Bayesian algorithm. With those semantic features, they ranked the basic highlight lever with the feature statistic and measured the quality of the game from a fuzzy system. To measure their tracking performance, they employed the two most commonly used criteria: recall (the percentage of the number of correct detections from the actual balls) and precision (the percentage of the number of correct detections in all the detected balls). The 2004 Olympic Female Final, 2000 Olympic Male Final and 2005 World Cup Male Final Games were selected to be the testing data with the resolution of (352*288) at 25 frames per second.

To summarise, the revision of the state-of-the-art research work regarding the developments in table tennis was presented. Even though they all were related with table tennis, different applications use different techniques for extracting the racket, detecting the ball and predicting its trajectory. Based on the above literature, feature-based approaches such as colour, shape, and size are appearing to be a common technique in most table tennis ball detecting mechanisms. It was followed by predicting the trajectory of the ball in track. Since the above literature is selected from recently published papers, the multiple cameras-based approach also appears to be the selected approach of most researchers, since it can provide the stereo vision with depth information. Section 2.4 will discuss some of the recognisable commercial deployed systems regarding ball detection and tracking for the support of decision making.

2.4 Commercially Deployed Systems

Computer vision technologies have been commercially deployed in tennis and football. Several systems have been used in real matches. The following subsections give an overview of such systems.

2.4.1 Hawk-Eye

Hawk-eye system is used in Wimbledon 2003 to produce a computer-generated replay which can help the commentary team to analyse the play in eight main areas (McIlroy, 2008). It is the camera-based ball-tracking system that enables players to challenge **line-calling decisions** on the courts of major tennis events. It is also known as one of the most commonly used technologies in various sports for effective decision making (Bal and Dureja, 2012).

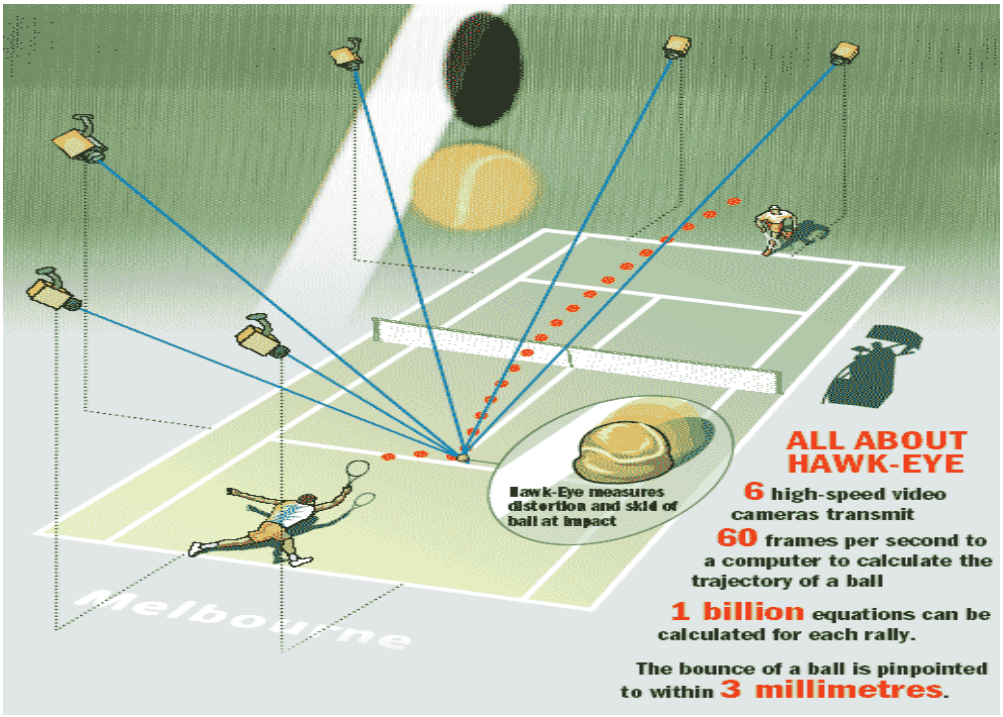


Figure 2.16: Hawk-Eye System (Tsang, 2013)

Six to ten high-speed cameras are situated around the court to determine where the ball has bounced within five seconds of landing and whether it is in or out (Fowler, 2012). The system has a mean error of 3.6 mm and is resistant to wind, sunlight, artificial floodlights and overcast conditions (Hawk - Eye, 2017). Hawk-Eye use the Direct3D interface in **Microsoft's DirectX API** to show the trajectory of the ball on a virtual court by rendering the graphics (Owens et al., 2003). Hawk-Eye Innovations become world-leading ball tracking technology for the sporting such as tennis, football, and cricket because of the capability of providing the virtual replay and providing the statistical viewable track within 5 seconds (Bal and Dureja, 2012) at the end of the rally.

After initial use at the Davis Cup in the UK, Hawk-Eye tennis was deployed at the Masters' Tournament in the United States and at the Queens tournament. (Owens et al., 2003) described a major design of the Hawk-Eye tennis ball tracking system as shown below in Figure 2.7. The system applied the 3D model-based tracking approach to determine the registration features and corresponding image observation. Firstly, it extracts straight lines by use of spatially adaptive thresholding. To deal with the issue of distortion, the KF is used to determine the impact point. KF takes in the tracking data directly and iteratively tries a linear, quadratic and cubic model of the incoming and outgoing compound track to get the best estimate of the impact point possible. For predicting the flight or trajectory of the ball, Hawk-Eye used a **Geometric algorithm** which employs a **triangulation process** for 2D position calculation and depth calculation (McIlroy, 2008). A rule-based system is used to decide whether an impact is a bounce, a strike or a half volley, using the velocity

directions and positions to reduce the degrees of freedom. A simple distance metric is used to determine which impact point is the most likely one. Hawk-Eye has been put to a variety of uses, such as providing a way to collect interesting statistics, construct illustrative visual representations of the gameplay and even helping viewers to better understand the umpiring decisions (Bal and Dureja, 2012).

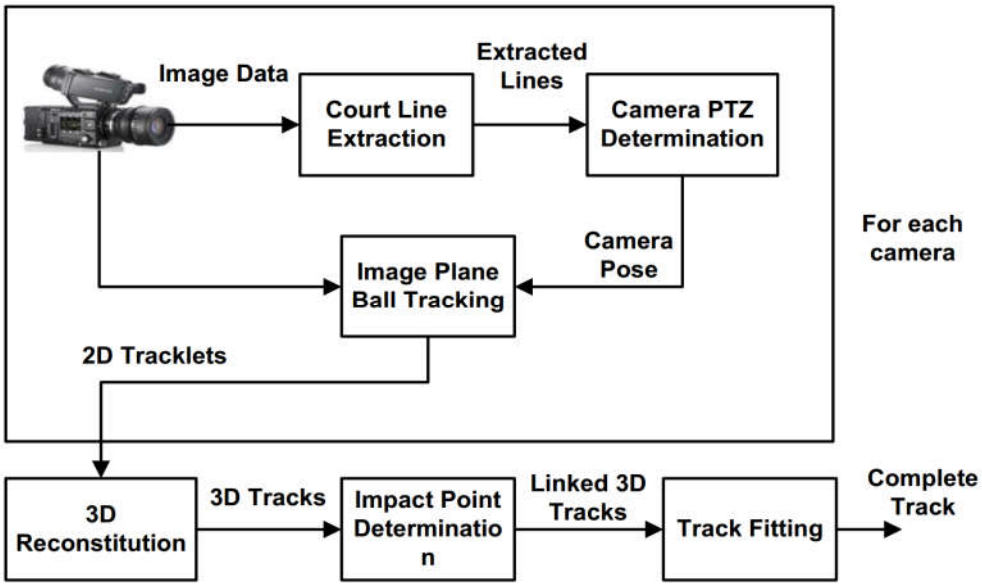


Figure 2.17: Block Diagram of Hawk-Eye tennis System (Owens et al., 2003)

2.4.2 Goal-line

Hawk-Eye has recently developed and tested a Goal-line technology (GLT) system to be used in football using the same technique as the systems seen in other sports (Fowler, 2012). GLT is six high-speed multi-cameras-based system used to determine when the ball has completely crossed the goal line with the assistance of electronic devices and at the same time, assisting the referee in

calling a goal or not. Starting from 2012, GLT is approved by the International Football Association Board (IFAB) as it meets the requirement of being accurate to +/-3cm and was used the first time at the 2012 FIFA Club World Cup in Japan (Hawk-Eye, 2018) . FIFA has recently confirmed GLT was used at the 2014 World Cup in Brazil (BBC, 2018).

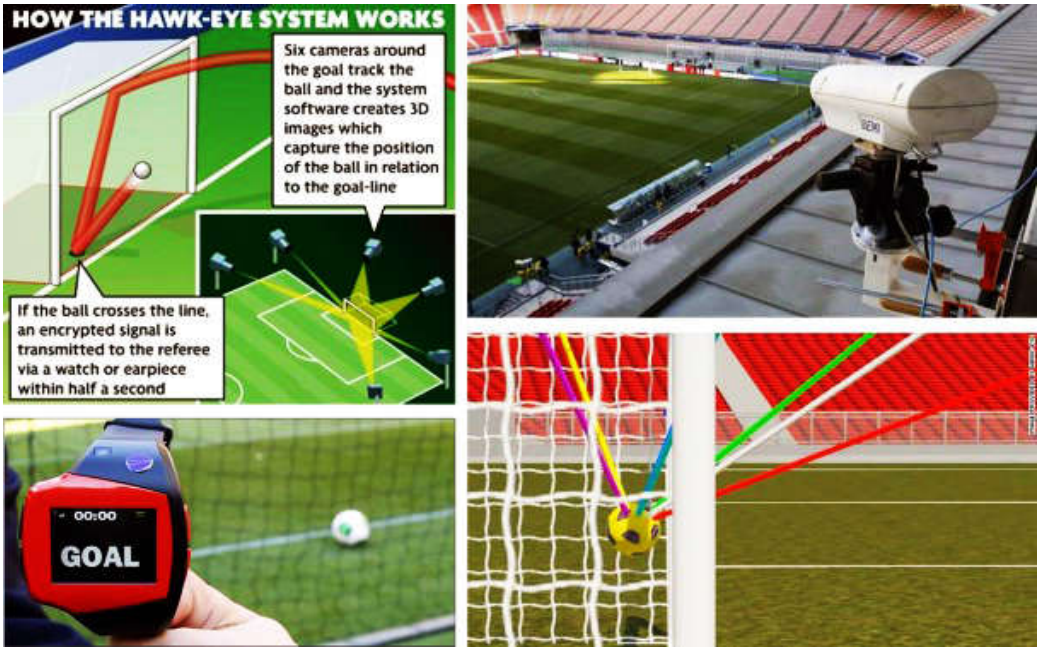


Figure 2.18: GLT (Sports live production, 2013)

2.4.3 GoalRef®

GoalRef® is a second GLT system. It has reached an agreement with FIFA to be installed in football grounds around the world (Fowler, 2012). GoalRef® uses a microchip embedded in the ball. When it crosses the goal line, it interrupts a magnetic field and signals a goal. Three magnetic strips are placed inside the outer lining of the ball, between the bladder and the outer casing, and when the ball crosses the line these are detected by sensors inside the goalposts and crossbar. The sensors send out electronic waves which are disrupted when

the ball crosses the line, and a computer then sends a message to the match officials' watch receivers in less than a second (Dovaston and Correspondent, 2012).

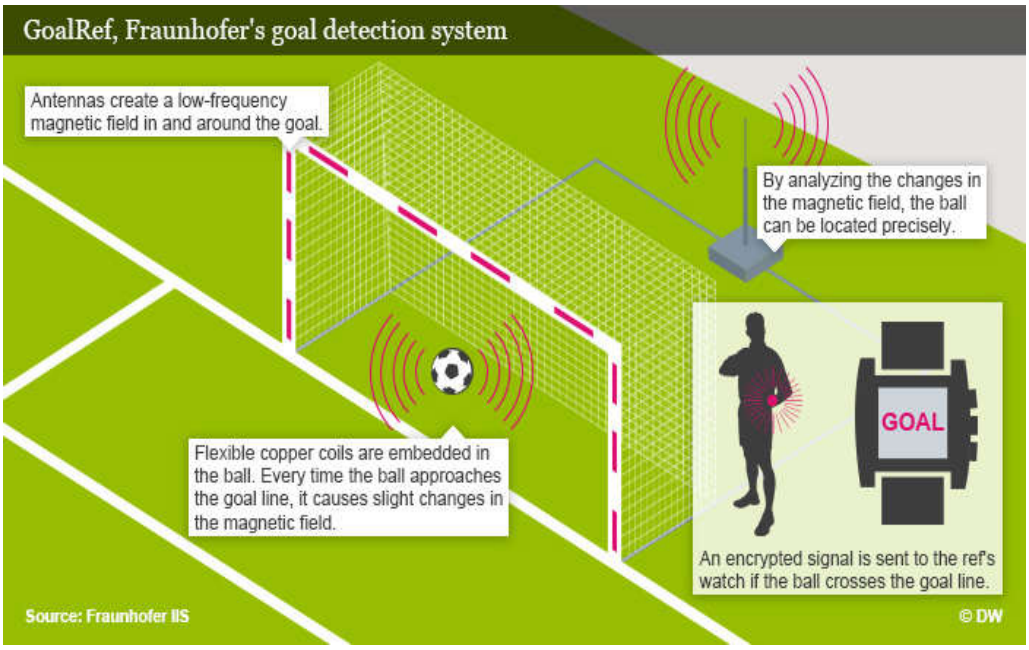


Figure 2.19: FIFA picks Goal Control for Brazil 2014 (FIFA, 2013)

However, there are a number of issues with Hawk-Eye and GLT. Firstly, the accuracy of both of the system is not 100%. There is still a need for an umpire, meaning human error has not been eliminated (Fowler, 2012). A second issue is a cost. To setup Hawk-Eye at a tennis tournament is about $60,000-$70,000 for one court (Hawk-Eye, 2018; Demaj, 2013) and with much of that cost going to installing the infrastructure. The cost of installation of the GLT is between £100,000 and £125,000 per stadium (BBC, 2018; Bal and Dureja, 2012). Although this is not an issue for **Premier League teams**, lower league teams may struggle to meet these costs.

2.5 Challenges and Possible Solutions

When detecting and tracking a fast-moving object, difficulties can arise due to abrupt object motion, changing appearance patterns of both the object and the scene, non-rigid object structures, object-to-object and object-to-scene occlusions, intrinsic and extrinsic factors like deformation and camera motion. Moreover, many environmental and capturing factors can influence the detection accuracy including:

- **Image Distortion:** When the object is travelling fast, if the shutter speed of the camera is not sufficiently high, the object can become blurred, colour faded and distorted in shape.
- **Multiple moving objects:** Apart from the target object, some other surrounded objects exhibit different motion.
- **Uneven lighting:** When detecting the object in an indoor environment, light sources are usually located in the ceiling, which tends to make the upper portion of an object appear brighter than the lower part.
- **Occlusion:** The target object can be blocked by the other objects or it can disappear from view.
- **Merging:** When the contrast between the object and background is low, the object may become indistinguishable from the background.
- **Object Confusion:** Background and foreground objects which have a similar colour, size, and shape may be confused with the target object.

- **Size:** If the target object's size is small, it will often only a few percent of the size of the frame, which renders conventional histogram-based detection methods unsuitable.
- **Time constraint:** If the development is a real-time application, the latency incurred for detecting and tracking the ball must be minimised which prevents computationally intensive algorithms from being adopted.
- **Cost constraint:** If the development is intended to be an affordable system, expensive equipment which provides higher precision are not employable.
- **Installation constraint:** If the development is intended to be a movable and easily deployable system, it will not have a facility to fix the cameras high above the detected area to take aerial views. Obtaining aerial views is not always possible as most table tennis tournaments take place at multi-purpose sports venue and fixing cameras at the ceiling or high wall is not allowed.

Among those challenges, one of the critical issues which come across is occlusion. Occlusion can happen when one part of the object occludes another (self-occlusion), when one or more objects occlude each other (inter-object occlusion) or when the target object becomes invisible from the camera viewpoints (out of camera-vision). Under occlusion conditions, the combination of different features such as colour, texture, shape, trajectory, speed, depth, etc can be taken into consideration.

2.5.1 Marker-based Solution

Object detection with the help of additional markers is another possible solution for the object merging problem. However, the proposed research is

intended to detect and track the table tennis ball from a real-world scene for umpiring purposes. During the match, players need to concentrate on the movement of the ball and it is not acceptable to apply a marker on the ball. Another disadvantage of this approach is the marker position on the object. The marker can only be detectable from a particular angle where the camera is set up. In reality, the spinning speed of table tennis ball can exceed 5000 rotations per minute (rpm) (Tamaki et al., 2012) and applying the marker on the table tennis ball for detection purpose is not practical.

2.5.2 Multiple Cameras based Solution

Early works (Tang et al., 2008; Wong, 2008; Chen and Zhang, 2006; Zaveri et al., 2004) were based on the use of a single camera. Later, researchers (Anuj and Krishna, 2017; Cheng et al., 2016; Takahashi et al., 2016; Liu et al., 2014; Bal and Dureja, 2012; Bao et al., 2012; Chen et al., 2010, 2011; Zhang et al., 2010; Arenas et al., 2009; Owens et al., 2003) started to consider the added value of using multiple cameras to enhance object detection. Employing multiple cameras can be more effective in tracking objects which are occluded from a specific viewing angle. With multiple cameras, the chance of capturing the ball is higher and it is no doubt a better approach. An additional benefit is that the depth estimation can be driven from two overlapping conjunction views and it can be used to predict the 3D trajectory (Bao et al., 2012; Chen et al., 2010, 2011). One example of multi-view ball tracking and virtual replays for tennis ball tracking can be seen in (Pingali et al., 2000). To minimize occlusions, six static cameras were placed around a stadium with four cameras on the side and two at

the ends of the court. Each of the four side cameras is paired with one of the end cameras to form a set of four stereo pairs that track the ball in 3D. To get the result in real-time, a multi-threaded approach was used which means each camera pair is associated with a computing device and works together as distributed computing resources. To cope with environmental changes and illumination effect, auto-iris lenses were used with the cameras in their approach.

However, synchronizing all networked cameras, maintaining continuous tracking, the huge quantity of incoming data increases computational overheads and makes the overall system complicated. Because of the varying lighting condition from different camera views and their positions, it is infeasible to ensure optimal monitoring among multiple cameras and synchronize all the detection processes by traditional approaches like human operators. In practice, developing a real-time tracking system needs not only a robust mechanism to detect but also it demands low computational algorithms to achieve the result as fast as possible. To solve this, one effective solution can be deploying a **Multi-Agent System (MAS)** which is often concerned with the coordination of autonomous agents to perform tasks, so that it can achieve high-quality overall system performance.

2.5.3 Multi-agent-based Solution

A MAS is composed of multiple interacting computing elements, which are systems that can decide for themselves what they need to do in order to satisfy their design objectives (Wooldridge and Jennings, 1995). Agents can also be defined as sophisticated computer programs which have a capability of

interacting with other agents and act autonomously on behalf of their users across distributed environments to solve a growing number of complex problems (Wooldridge, 2008). By this way, each agent can determine relative positions and orientations of all other agents performing tasks in its field of view. With the help of MAS, many observations can be made within a short period of time and can achieve the umpiring result in real-time. MAS are an increasing trend, used in various systems (Anuj and Krishna, 2017; Cañizares et al., 2017; Chakroun et al., 2011; Yongping Li et al., 2010; Katalenic et al., 2009) such as the detection of events in the sports video, segmentation and moving object tracking applications.

A considerable amount of research effort (Shao and Xie, 2012; Swears and Hoogs, 2012; Chakroun et al., 2011; Katalenic et al., 2009; Chao and Jun, 2008; Jin et al., 2006) has recently been focused on the cooperation of artificial intelligence systems in object detection and tracking. Their systems are implemented through detection algorithms that process images captured by cameras mounted on multiple computing agents that interact in a time-varying manner. In this way, each agent can determine relative positions and orientations of all other agents performing tasks in its field of view that can cover the occlusion.

2.6 Artificial Intelligent Systems

Research in **Artificial Intelligent (AI)** system is directed toward building a machine that can mimic or exceed human mental capabilities, including

reasoning, understanding and recognition (Hopgood, 2012). AI allows computer system to function in an intelligent manner which can learn on its own.

2.6.1 Techniques of Intelligent Systems

AI systems can be seen as three major types; computational intelligence systems, knowledge-based systems, and hybrid systems.

- **Computational intelligence (CI)** includes artificial neural networks, genetic algorithms and other optimization algorithms which can learn for themselves from a set of data.
- **Knowledge-based systems (KBS)** include expert and rule-based systems, object-oriented and frame-based systems, and intelligent agents.

Artificial Neural Network (ANN) (Caudill, 1987) is a computing system made up of a number of simple, highly interconnected processing elements, which process information by their dynamic state response to external inputs (Gerven and Bohte, 2018). ANN can be used to train a system by adjusting weight according to the input data. Although ANN can mimic the human brain and good at solving complex problems, it requires a huge amount of data and very computationally expensive to train.

Genetic Algorithms (GAs) (Mitchell, 1998) are adaptive heuristic search techniques used for finding optimized solutions to search problems. Although GAs mimic biological evolution and can provide an optimal solution which can scale well (adjust) to higher dimensional problems, it is sensitive to the initial numerical populations used, require heavy simulation and slow in producing results.

The **KBS** (Grzegorz J. Nalepa, 2018) contains thousands of complex rules and facts in which those are represented explicitly and can be changed at will. Rules are an effective way of representing knowledge in many application domains. Although both of expert and rule-based systems are types of KBS, in which the knowledge is represented in a form of sets of rules, expert systems can offer advice, suggestions, or recommendations like a human expert. A key advantage of the rule-based system is their flexibility. In a fully automated system, the rules can perform their recommended actions rather than simply making a recommendation. The principle difference between a KBS from a conventional program is in KBS, it is easier to add new knowledge, either during program development or in the light of experience during the program's lifetime.

The **Object-oriented programming (OOP)** (Danny Poo, 2008) languages such as C++ or Java are widely used in the development of object-oriented and frame-based systems. Both techniques assist the system by breaking down complex problems into simpler components, more maintainable, adaptable and reusable while maintaining the integrity of the system.

Intelligent Agents (Kohei Arai, 2018; Wooldridge and Jennings, 1995) extend the ideas of object-oriented techniques. When a software system becomes larger and very complex to maintain as centralised system, the whole system can be broken down as modules and turn into agents which can make their own decisions based on its own experience and circumstances. Intelligent Agents are a software system that can learn through experience, adapt over time, response to the demands, provide reasoning capabilities and take care of specific tasks to meet their designed objectives.

A **multi-agent system (MAS)** (Wooldridge, 2008) is a computerized system composed of multiple interacting intelligent agents which are working together to solve problems that are difficult for an individual agent (single hardware or software) to solve. In MAS, agents can cooperate (or competitive) and negotiate with other agents, yet each agent can autonomously make its own decision to achieve their designed goals. It means agents cannot invoke the actions of another agent, but they can make requests. As a teamwork, MAS offers the benefits of;

- Automation
- Speed and efficiency
- Robustness
- Reliability and consistency
- Ease in development
- Scalability
- Cost-effective

than a large centralised system.

2.6.2 Agent Tools and Languages

The tools available to assist in developing intelligent systems can be divided into the following categories:

- Agent-based modelling software such as **NetLogo, JADE**
- Stand-alone packages such as **expert system shells**
- KBS toolkits such as **Flex**
- AI programming languages such as **Lisp**

- Libraries such as **MATLAB**
- OOP languages such as **C++, Java**
- Conventional programming languages such as **C**.

Agent Communication Language (ACL), proposed by the Foundation for Intelligent Physical Agents (FIPA | IEEE, n.d.), is a proposed standard language for agent communications. The most popular ACLs are:

- **FIPA-ACL** (by the FIPA, a standardization consortium) (Poslad, 2007)
- **KQML** (Knowledge Query and Manipulation Language) (Finin et al., 1994)

Agent-oriented programming (AOP) is a programming paradigm where the construction of the software is entered on the concept of software agents. **OOP** languages are based on **AOP** because of the clear separation of function, structure, and state. For the Java-platform one of the frameworks which meet the standard of FIPA is **JADE**.

2.6.3 Application of Intelligent Systems

In science, technology, and engineering, a large number of an intelligent system has been developed recently by using many different techniques. Multi-agent systems have been applied in soccer forecasting (Cañizares et al., 2017), smart surveillance system (Eigenraam and Rothkrantz, 2016), decision-making system for sporting event (McNeill et al., 2016), social media sentiment detection such as Facebook and Twitter (Charaf et al., 2012), moving object tracking (Chakroun et al., 2011), sport event detection (Yongping Li et al., 2010), in line detection in sport images (Kamarposhty et al., 2009) and many other research areas. Among those, applying the multi-agent system in sports have substantially

grown during these years due to the requirement of real-time recognition of events in sport. Since each agent can take over an assigned task, a complex system can be divided, and a powerful collaboration of intelligent agents can increase the overall performance.

2.7 Summary

In brief, to achieve a high success rate in fast-moving object tracking, the main problems are lack of unique features, blurred motion, the complexity of background, easily affected by luminance, and multiple moving objects which make it difficult to distinguish the ball from its surroundings. To explore new ways to solve these problems, it is necessary to critically analyse and discuss an intensive survey of object detection literature. Therefore, a large variety of existing approaches for object segmentation, detection and tracking have been presented in the first part of this chapter. Merits and demerits of available methods have been discussed in detail as a comparative study. The second part of this chapter gave an overview of the computer vision technologies which have been employed in various sports. Although many publications related to ball tracking are available, the techniques that produce satisfactory results are usually object and application specific. It is, therefore, the survey of literature for this study focuses on tracking table tennis ball. The intensive survey reveals that two paper discusses tracking for umpiring purposes (Wong and Dooley, 2010) and (Byrd, 2015), while all the other papers discuss the development of playing robots. While satisfactory results were achieved, (Wong and Dooley, 2010) only covered the service part of the rallies, of which the ball is not traveling at very

high speed. On the other hand, an automatic scoring system was discussed by (Byrd, 2015) that tracked the table tennis ball in real-time relative to the table and net, and determined when a point is scored. However, the setting is based on a lab environment with the ball painted neon green against a uniformly black background. Such a setting significantly reduces the ball detecting difficulty, but it will not be acceptable for formal tournaments as the Law of Table Tennis mandates the colour of the ball to be either matt white or orange (ITTF, 2018). In summary, none of the literature reviewed addresses the challenges of tracking the ball in a complete rally and in a real match scene, which is an essential requirement for a realistic automatic umpiring system. It was followed by the revision of the state-of-the-art research work regarding the technological development in table tennis. The last part of this chapter discusses the challenges that are frequently faced in object detection and tracking research and provides possible solutions including the **AI** systems.

Chapter 3

Research Methodology

3.1 Introduction

This chapter presents the research methodology adopted to design, develop as well as test and critically evaluate the new multi-agent system for video umpiring. Automatic umpiring systems are very complex in nature due to the architectures and the match environment in which they are deployed. The accuracy of 3D information is crucial when identifying the location of objects from video for umpiring purposes. This chapter presents the development methodology and tools used in assessing the overall performance of the proposed system. The remainder of the chapter is organised as follows: Section 3.2 provides the research methodology while Section 3.3 explains about camera set up. Section 3.4 and 3.5 describe building a research testbed and test sequences. Section 3.6 presents the evaluation methodology and finally, Section 3.7 summarise the chapter.

3.2 Research Methodology

As mention in Chapter 1, table tennis has a myriad of diverse rules governing the legality of a rally and there are many technical challenges in developing a low-cost umpiring system. This combination of system requirement,

complexity and challenging variable environments leads to the design and analysis of problems that are not analytically tractable by applying existing toolkits. It requires a completely new testbed and a real implementation to be developed from scratch to investigate the feasibility the of system. The outcome is that it provides a valuable insight into system behaviour as well as performance and can identify the limitations. To meet the objectives which are set out in Section 1.3, the various phases of the adopted research methodology to fulfil the aim of automatic table tennis match umpiring are summarised as follows:

- Critical review of object detection and tracking literature as well as AI systems and narrow down the research focus to ball detection and tracking.
- Implement a test bed for experimenting variety of new techniques and investigate the suitability and expandability of the existing methodologies.
- Identify the limitations of existing methods and develop a new algorithm to fulfil the system's requirements.
- Implement a ball tracking system which will be used as an underlying building block for the development of an umpiring system.
- Rigorously test and critically evaluate the developed system for different table tennis sequences captured from real match scenes to validate its accuracy and robustness.
- Compare the performance of the system against the ground truth provided by human umpires.

3.3 Camera Set up

When the aim is a cost-effective system, the capturing devices should be buildable from affordable and off-the-shelf materials. It should also have speed and resolution that are high enough to capture the most important moments such as ball's bounce or hit points. However, high-speed stereo cameras are expensive. Therefore, it was decided to utilise existing low-cost cameras from the Open University with acceptable capturing speed to create in-house stereo systems to balance between cost and accuracy of detection. In the course of this research, several table tennis sequences were filmed using up to 4 Casio EX-F1 cameras during table tennis matches. Three possible types of camera setup were experimented as follows:

- Side-By-Side Stereo-view (Full-Table)
- Side-By-Side Stereo-view (Half-Table)
- Face-To-Face Multi-view (Half-Table)

3.3.1 Configuration of Side-By-Side Stereo-view (Full-Table)

For the Side-By-Side (SBS) arrangement, two set of stereo visions (full-table and half-table) are formed by pairing up two cameras next to each other and fixing them on camera mounts as shown in figures 3.1 and 3.2. In this way, similar results as taken by stereo cameras can be achieved while reducing the cost. Stereo vision is required rather than single vision because it can provide depth information from two overlapping conjunction views and can derive the 3D

trajectory. When capturing the sequences, the camera pairs are positioned at the location where the umpires would sit as shown in figures 3.1.

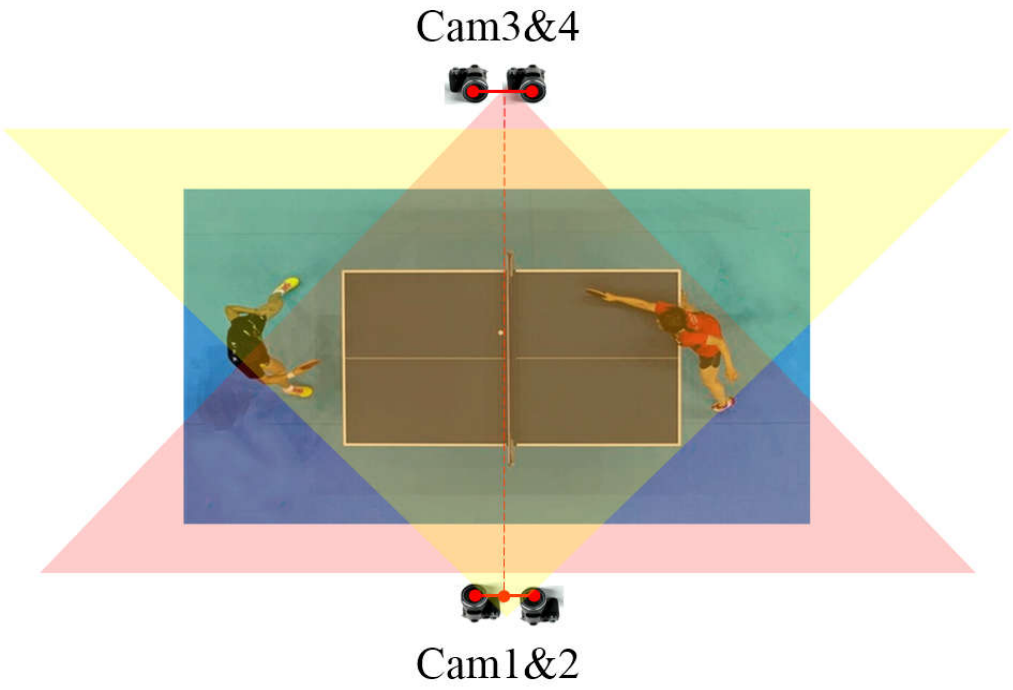


Figure 3.1: SBS Stereo Camera (Full-Table) Set up

In this setup, a pair of stereo cameras monitored the full length of the table at both sides. To achieve the whole table view, the constraint here is that the camera needs to be placed at a distance from the play (greater than 4 meters). As a result, the videos which were filmed from a long distance, do not have enough depth resolution to calculate precise real-world coordinates (X, Y, Z) of objects and this affected 3D reprojection. The accuracy of 2D to 3D reprojection become poor when the resolution of the camera is low, and the object is further away from the cameras. Moreover, the original size of table tennis ball is small, only 4 cm in diameter, and when capturing the full table view, the video image of the ball appears very small, 1.9 pixels, 0.005% of the whole frame. As larger image size

of the ball and clearer view of the play can provide higher detection rate, it is better to place the cameras at a closer position.

3.3.2 Configuration of Side-By-Side Stereo-view (Half-Table)

To resolve this, two pairs of stereo cameras were placed next to each other to monitor half the length of the table from the same side. Since they can capture from closer to the area of play (approximately 2 meters), the image of the ball appears bigger and clearer. The average radius of ball is around 3 pixels in the film and the result of 2D to 3D reprojection result is sufficient to umpire a rally. However, the trade-off of relocating these cameras to closer positions is that one pair can only capture approximately two thirds of the length of the table not the whole view. To cover the whole table, one pair monitors the left-half of the table while another pair monitors the right-half as shown in figure 3.2. This arrangement requires a mechanism to track a trajectory using join data from separate views. Moreover, it is undeniable that the ball can sometimes be occluded. For this scenario, the SBS cameras arrangement cannot provide much help to each other in ball detection and recovering mechanism as both cameras are placed next to each other, much of the same view and getting similar information.

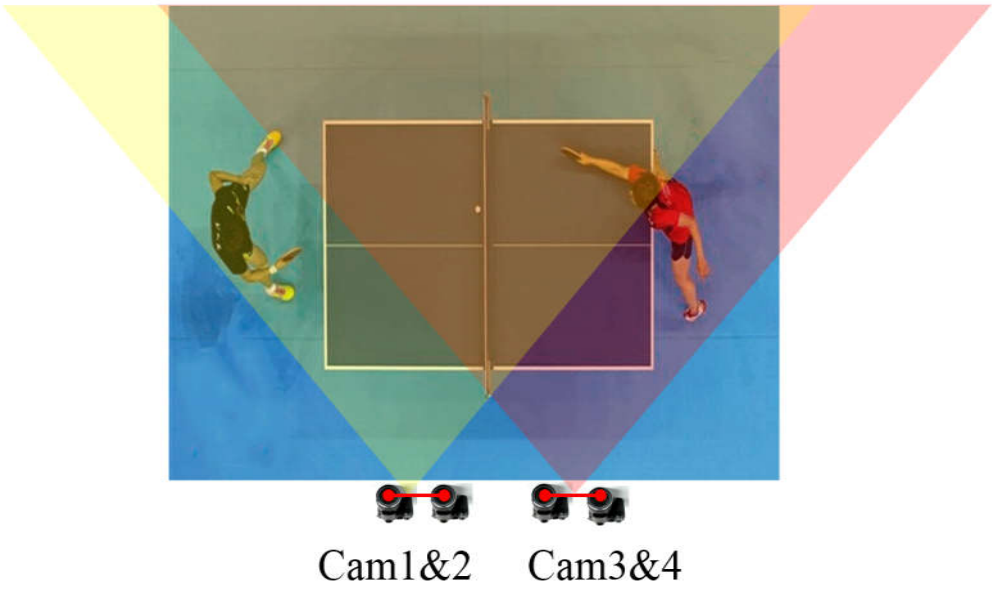


Figure 3.2: SBS Stereo Camera (Half-Table) Set up

To resolve this, this research proposed a new camera arrangement which is named Face-To-Face (FTF) to detect the 3D position of the ball using opposite facing cameras.

3.3.3 Configuration of Face-To-Face Multi-View (Half-Table)

In FTF arrangement, two pairs of opposite facing cameras were monitoring approximately half of the table. This is a new way of forming a stereo camera by pairing of opposite facing cameras. The trade-off of relocating these cameras to closer positions is they can only cover approximately two thirds of the length of the table, not the whole view. To cover the whole table, one pair monitors the left-half of the table while another pair monitors the right-half as shown in figure 3.3. As each individual camera does not have to cover the entire

table, the cameras can be placed closer to the objects of interest (e.g. ball and table) so that better depth resolution can be achieved when deriving their 3D positions and the objects also appear bigger in the views.

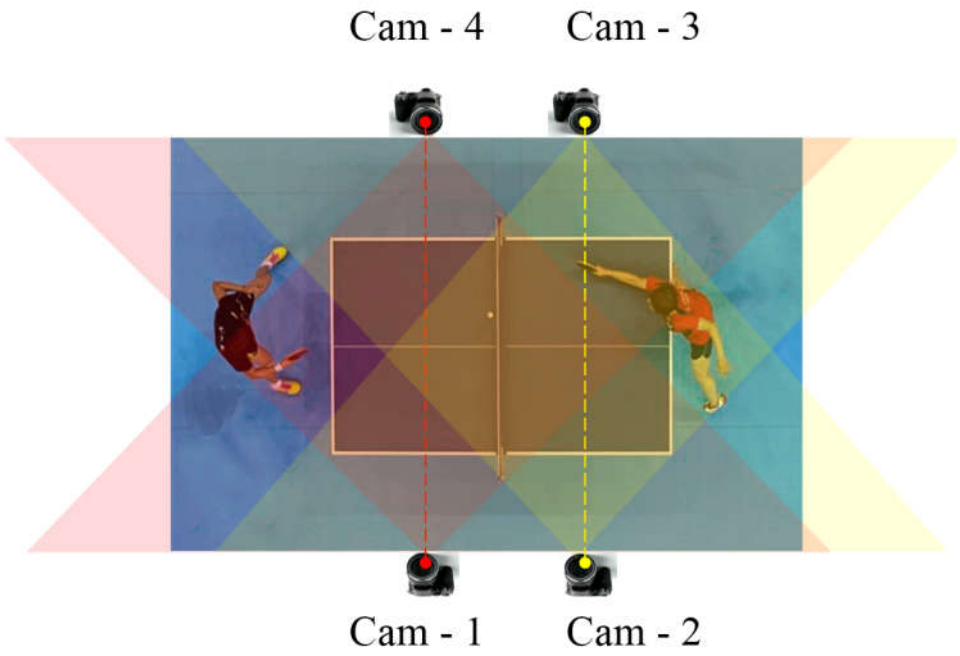


Figure 3.3: FTF Stereo Camera (Half-Table) Set up

However, this arrangement requires a mechanism to join a trajectory from separate views (the contributed method can be found in Chapter 5). Moreover, the drawback of the FTF configuration is that when the position of the ball is at or near the line that joins the principal points of the opposite facing cameras (see the red and yellow dotted lines in Figure 3.3), the 3D position cannot be determined using the triangulation equations, as the angles between the ball and the two cameras are equal to or close to zero. To overcome this problem, the position of the ball in these small regions will be extrapolated using a second order equation of motion, which can model the trajectory of the ball in a table tennis rally appropriately.

3.3.4 SBS Stereo Camera Calibration and 3D derivation

One of the basic requirements in creating a stereo vision is to calibrate the two cameras in order to achieve the parameters that need to be used in derivation of 3D information. The calibration process aims to correct camera distortions such as the lens and transitional errors. It can be conducted in two ways. The first way is to check and adjust the cameras' hardware and positions by using high precision optical tools which are expensive and can only be operated by trained professionals. The second method is to use software to adjust the captured images from the cameras. In this research, the software calibration with chessboard approach (Bradski and Kaehler, 2008a) was selected to compensate for the distortions because the corners of the squares in the chessboard are very easy to find as reference points by using computer vision algorithms and its geometry is very simple. It is also inexpensive and a widely used method by the research community. The figure 3.4 (a) and (b) shows the images of a chessboard being held at various orientations to provide enough information to completely solve for the locations of those images in global coordinates (relative to the camera) and the camera intrinsic. The intrinsic matrix contains the focal length expressed in pixel-related units and the principal point which locates the image centre, while the extrinsic matrices contain the rotation matrix and translation vector (Bradski and Kaehler, 2008a).

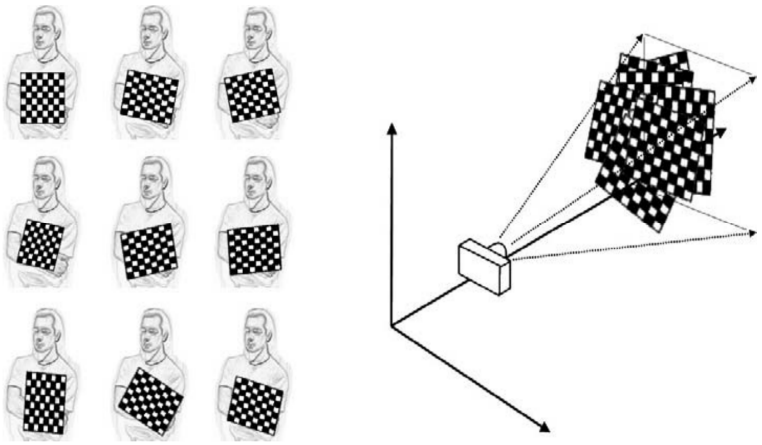
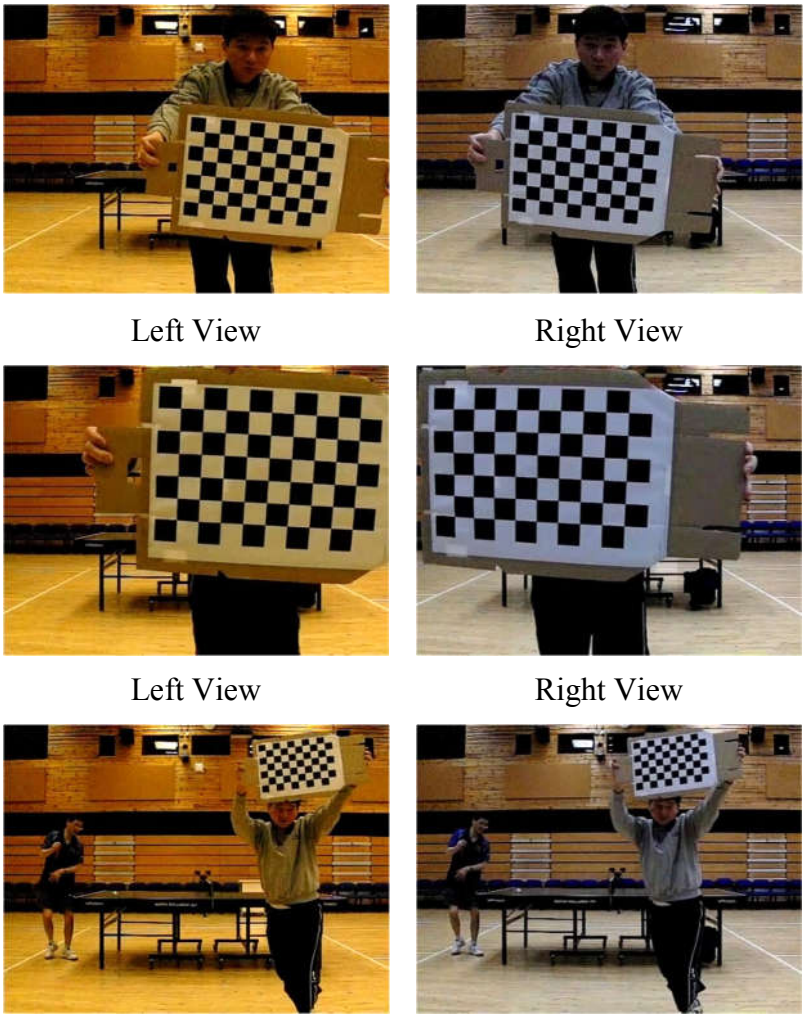


Figure 3.4 Chessboard calibration approach (Bradski and Kaehler, 2008a)



Left View Right View

Left View Right View

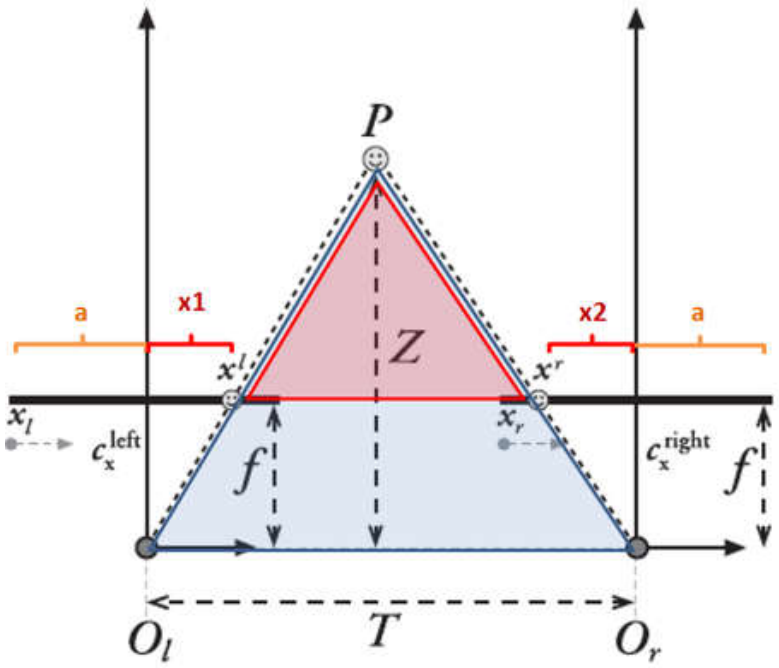
Left View Right View

Figure 3.5: Placing checkerboard at different locations

In software camera calibration, the first step is to correct the distortion caused by imperfection of the lens and its arrangement, while the second step is to correct the correspondence errors between two cameras. To correct lens distortion, a sequence of known patterns is first presented to the cameras. The software then mathematically creates an ideal pin-hole camera and distortion models based on the positions of a set of detected reference points. By evaluating the deviations between the detected and the known positions of the reference points, it is possible to derive the extrinsic and intrinsic matrices which can be used to correct the errors. These matrices are used for squeezing, stretching, rotating and translating different areas of an image in order to minimise the deviations between the detected and the known positions of the reference points. These parameters enable the software to mathematically correct the abovementioned errors. To summarise, stereo camera calibration involves:

- **Undistortion:** Mathematically remove the radial and tangential lens distortion of the cameras.
- **Rectification:** Adjust the images such that the angles of the cameras are aligned. The outputs of this step are images that are row-aligned and rectified.
- **Correspondence:** Find the same features (the reference points) in the left and right camera views. Assuming the camera pair are horizontally aligned, the output of this step is a disparity map *(d),* where the disparities are the differences in X-coordinates on the image planes of the same feature viewed in the left and right cameras: $d = x_l - x_r$

- **Reprojection:** If the focal length of the cameras and the distance between them are known, then a depth map can be derived from the disparity map using the triangulation theory as shown in figure 3.6 below.



$$x^l = a + x1 \quad (1)$$

$$x^r = a - x2 \quad (2)$$

$$d = x^l - x^r = x1 + x2 \quad (3)$$

Where:

a: half of screen width

x1, x2: the variable

x^l, x^r : the horizontal positions of the points in the left and right images

c_x^{left}, c_x^{right} : the principal points

O: the centre of projection

T: the total length bet two Cameras: cm

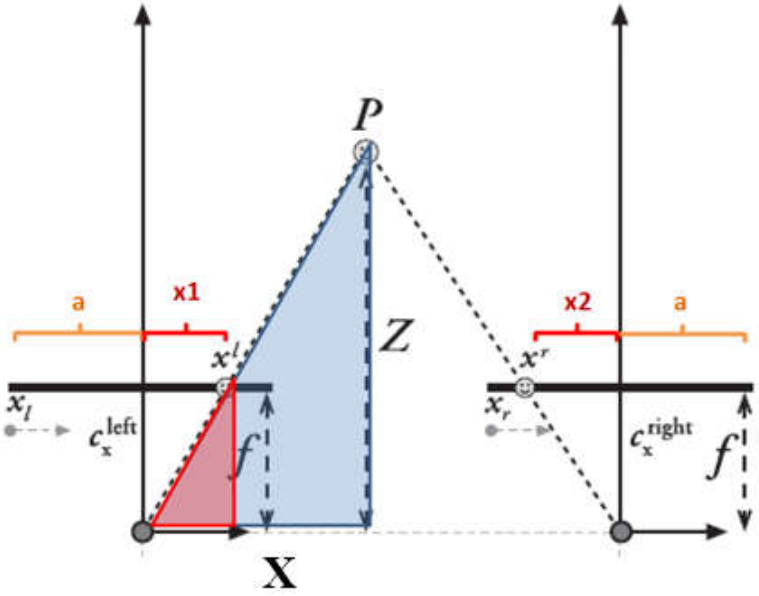
f: the focal length of the camera.

d: disparity

Z: the depth

Figure 3.6: Triangulation theory for Z (Depth) derivation

$$\frac{T}{T-(x1\ +\ x2\ )} = \frac{Z}{Z-f}$$
$$\frac{T}{T-(x^l - x^r\ )} = \frac{Z}{Z-f}$$
$$T\ (Z-f) \quad = \quad Z\left(T-(x^l-x^r)\right)$$
$$TZ - Tf \ = \ TZ - Z(x^l - x^r)$$
$$Tf \ = Z(x^l - x^r)$$
$$Z = \frac{Tf}{x^l - x^r} \tag{4}$$



Where:

a: half of screen width (512/2)

x1, x2: the variable

x^l, x^r : the horizontal positions of the points in the left and right images

c_x^{left}, c_x^{right} : the principal points

O: the centre of projection

T: the total length between two cameras: cm

f: the focal length of the camera.

Z: the depth

Figure 3.7: Triangulation theory for X and Y derivation

According to the equation (1), the 3D position of X and Y can be derived by using the following triangulation and geometry calculations based on the blue and red similar triangles in figure 3.7.

$$x1 = x^l - a \tag{5}$$

Based on equation (5), *a* is a half of the screen width (For example: 512 pixels/2).

In this research, the table tennis sequences were filmed using up to four Casio Exilim Pro EX-F1 cameras with their built-in lenses set to the widest angle (36mm). The cameras come with a 1/1.8" (0.7144 cm Width x 0.5358 cm Height) CMOS sensor (Casio Exilim Pro EX-F1 Sensor Info & Specs, n.d.). To calculate the real-world 3D position of an object, its screen position (in pixel units) needs to be converted into a physical unit such as centimetre. This can be achieved by dividing its screen position by its sensor dimension, i.e., (horizontal position/0.71) and (vertical position/0.54). The resolution of the frame captured by the sensor is (512 x 384) pixels. Therefore, there is 716.69 pixels in one cm (512 pixels / 0.7144 cm = 716.69 or 384 pixels / 0.5358 cm = 716.69) for the Casio EX-F1 cameras.

$$\frac{Z}{f} = \frac{X}{x1}$$

$$X = \frac{Z*x1}{f}$$

$$\mathrm{X} = \frac{\mathrm{Z}*(x^l - \mathbf{a})}{f}$$

$$\mathrm{X} = \frac{\mathrm{Z}*(x^l - \frac{512}{2})}{f}$$

$$\mathrm{X} = \frac{\mathrm{Z}*\left((x^l - \frac{512}{2})/716.69\right)}{\mathrm{f}} \tag{6}$$

Based on the similar triangles' rules with the Y-axis perpendicular to the page:

$$\frac{\mathrm{Z}}{f} = \frac{\mathrm{Y}}{\mathrm{y1}} = \frac{\mathrm{Y}}{\mathrm{y^r}}$$

$$\mathrm{Y} = \frac{\mathrm{Z}*\mathrm{y^r}}{f} \tag{7}$$

Here, assuming two cameras whose image planes are exactly coplanar with each other, and exactly parallel optical axes with equal focal length, f. The optical axis is the ray from the centre of projection, *O* through the principal point, *c* which is where the principal ray intersects the imaging plane. A point, *P* in the physical world is projected in the left and the right image views at x^l and x^r. The disparity *(d)* between these views is defined simply by $d = x^l - x^r$. Since depth is inversely proportional to disparity, the depth, $\boldsymbol{Z}$ can be derived by using triangulation theory based on the blue and red similar triangles in figure 3.6.

3.3.5 FTF Stereo Camera Calibration and 3D derivation

For calibration purposes, a checkerboard was carefully placed at various known positions during filming as shown in Figure 3.8 and 3.9. This process provides a large set of reference points needed for training the error model which can be found in Chapter 5. The set of reference points in the checkerboard was also used for results comparison to assure that the system can provide the right 2D to 3D reprojection results.

Figure 3.8: Calibration with a double-sided checkerboard

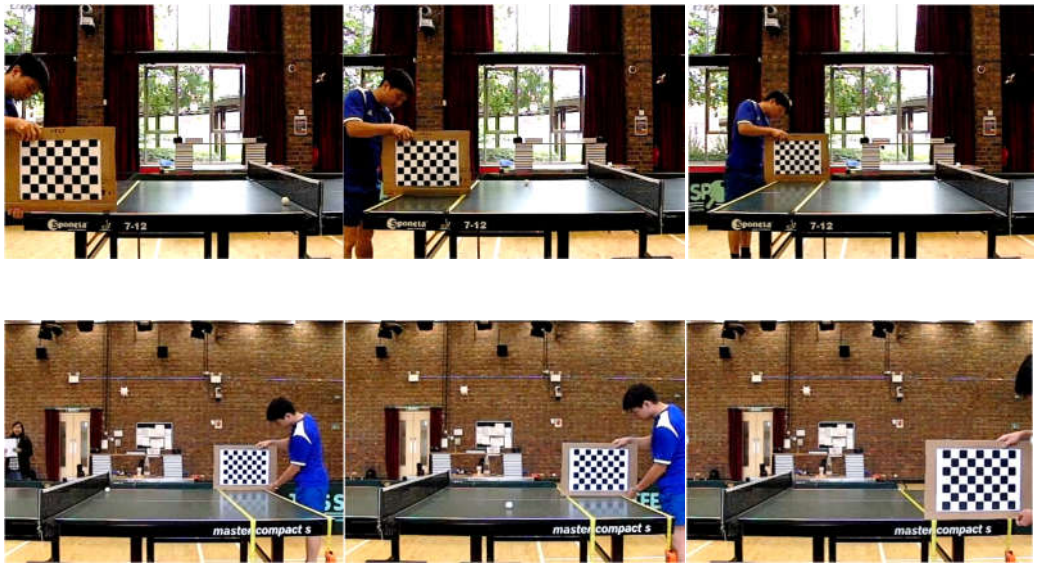


Figure 3.9: Placing double-sided checkerboard at different location

The 3D positions derived from the FTF camera pair are computed using the following triangulation and geometry calculations as shown in figures 3.10 and 3.11.

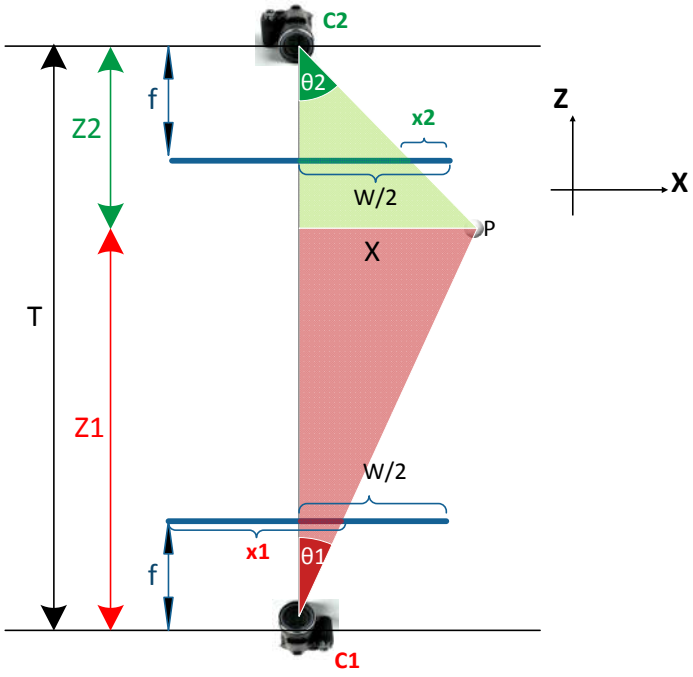


Figure 3.10: The aerial view of FTF camera pair

The *X* and *Z* values of the 3D ball position in the FTF configuration can be derived as follows:

$$\begin{aligned} T &= Z_1 + Z_2 \\ &= \frac{X}{\tan \Theta_1} + \frac{X}{\tan \Theta_2} \\ &= X.\left(\frac{\tan \Theta_2 + \tan \Theta_1}{\tan \Theta_1 . \tan \Theta_2}\right) \\ X &= T.\left(\frac{\tan \Theta_1 . \tan \Theta_2}{\tan \Theta_1 + \tan \Theta_2}\right) \end{aligned} \quad (8)$$

where *X* is the value of horizontal 3D position of the ball (Distance), which can be derived from *T*, the distance between two cameras and the unknown values

$tan\,\Theta_1$ and $tan\,\Theta_2$. These unknown values can be found by equations (9) and (10):

$$tan\,\Theta_1 = \frac{\left(x_1 - \frac{W}{2}\right)}{f} \tag{9}$$

$$tan\,\Theta_2 = \frac{\left(\frac{W}{2} - x_2\right)}{f} \tag{10}$$

where $\tan\Theta_1$ and $\tan\Theta_2$ can be calculated by the triangulation and geometry calculations. In which, the known values x_1 and x_2 are the screen coordinate of the centre of the ball in pixels, W stands for the screen width and f is the focal length of the cameras. After that, the 3D position of Z can be calculated by (11) and (12):

$$Z_1 = \frac{X}{\tan\Theta_1} \tag{11}$$

$$Z_2 = \frac{X}{\tan\Theta_2} \tag{12}$$

where Z_1 and Z_2 indicate the associated distance between the ball and each camera (Depth). Similarly, the Y value of the 3D ball position in the FTF configuration can be calculated as shown in figure 3.9.

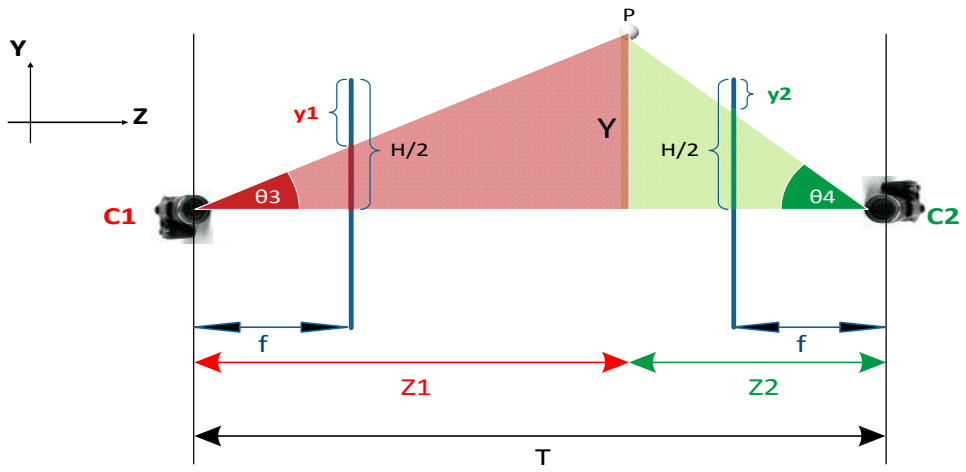


Figure 3.11: The side view of FTF camera pair

$$
\begin{aligned}
T &= Z_1 + Z_2 \\
&= \frac{Y}{\tan\Theta_3} + \frac{Y}{\tan\Theta_4} \\
&= Y \cdot \left(\frac{\tan\Theta_4 + \tan\Theta_3}{\tan\Theta_3 . \tan\Theta_4}\right) \\
Y &= T \cdot \left(\frac{\tan\Theta_3 . \tan\Theta_4}{\tan\Theta_3 + \tan\Theta_4}\right) \qquad (13)
\end{aligned}
$$

where Y is the value of vertical 3D position of the ball (Height) which can be derived from the known value T, the total length between two cameras and the values $\tan\Theta_3$ and $\tan\Theta_4$ which can be found by equations (14) and (15):

$$\tan\Theta_3 = \frac{\left(\frac{H}{2} - y_1\right)}{f} \qquad (14)$$

$$\tan\Theta_4 = \frac{\left(\frac{H}{2} - y_2\right)}{f} \qquad (15)$$

As above, $\tan\Theta_3$ and $\tan\Theta_4$ can also be calculated by the known values y_1 and y_2 which are the screen coordinates of the centre of the ball on screen in pixels, H stands for the screen height and f is the focal length of the cameras. The calculated 3D results of the reference points on the net and table are shown in figures 3.12, 3.13 and 3.14. To synchronize all the (X, Y, Z) values of four cameras, one central point (one side of the top of the net) was set as an origin (0, 0, 0) and that point is named as the Net Pole Origin (NPO).

Figure 3.12: Calculated 3D results based on Camera 1 and 4

Figure 3.13: Calculated 3D results based on Camera 2 and 3

Figure 3.14: Calculated 3D results of reference points on table

To verify the 3D derivation, the height and length of the net, and the length and width of the table were calculated based on the formula and compared with the actual measurements. Figures 3.15, 3.16 and 3.17 show 2D to 3D reprojection results. The average error is less than 1 cm. However, it is noticed that the length of two reference points, e.g. the height of the net, derived by the two camera pairs are slightly different. These small differences are due to the slight misalignments between opposite facing cameras and measuring errors. This problem will be addressed in Chapter 5 and the detailed discussion of error correction method can be found in Section 5.2.

Figure 3.15: Calculated 3D results for the Net (Height)

Figure 3.16: Calculated 3D results for the Net (Length)

Figure 3.17: Calculated results for reference points on table

3.4 Testbed Development

The testbed is designed to take videos directly from Universal Serial Bus (USB) based cameras or from recorded video files for repeated testing. In this way, it has the flexibility in experimenting with different video sources such as live-fed video images. All experiments were conducted on a computer with an Intel® Core™ i7 CPU @ 2.80 GHz. To fulfil the aim of research, it is required to implement three essential developments:

- Ball Detection Algorithm development
- Multi-agent Umpiring System development
- Bridge mechanism between Ball Detection Algorithm and Multi-Agent System.

The ball detection algorithm was developed for detecting the 3D location of the ball and the multi-agent umpiring system was developed for monitoring the status of the rally and determining when a fault occurs. The basic framework of the testbed was developed to accept either a SBS or FTF stereo camera configuration for experimenting with different ideas and evaluating the system.

3.4.1 Selecting Platform and Tools

For Ball Detection Algorithm implementation, C++ was selected as the programming language as it provides facilities for low-level memory manipulation and is widely used in performance-critical applications. For computer vision research, the two most widely used tools are Open Source Computer Vision Library (OpenCV library 2017, n.d.) and Matrix Laboratory (MATLAB) (MathWorks, 2018, n.d.). OpenCV is a library of programming functions mainly aimed at real-time computer vision and image processing tasks. It was originally developed by Intel and subsequently became open-source. MATLAB is a matrix based general purpose mathematical tool with a programming environment. It is widely used by the industry and research communities. OpenCV was chosen for developing this research test bed because of the following reasons:

- **Cost:** MATLAB costs about $2,000.00 per license. OpenCV is free, and it is an open source computer vision and machine learning software library.

- **Speed:** The source code of OpenCV is designed to be light weight with high computational efficiency. It has a strong focus on real-time applications and has been optimised to run faster.
- **Functionality**: OpenCV provides more functions for image and video processing than MATLAB does.

For the development of the Multi-Agent Umpiring System**,** Agent Based Modelling and Simulation (ABMS) tools were used instead of modelling with conventional programming tools which requires burden of housekeeping tasks such as memory management and synchronization mechanism. Among the most widely employed ABMS tools such as Swarm, Repast, NetLogo and Java Agent DEvelopment Framework (JADE) (Anon, 2017), JADE was chosen in terms of following reasons:

- **Standardisation:** JADE complies with the de facto standard set by FIPA (FIPA | IEEE, n.d.).
- **Expendability:** JADE can be distributed across a network of computers and agents can be migrated from one machine to another if required.
- **Cost:** JADE is an open-source, platform independent and widely used for agent frameworks.

In developing separate solutions with two different integrated development environments (IDE), one challenge here is to bridge the two separately developed programs, the Ball Detection Algorithm in C++ and the Multi-Agent System in Java. One easiest solution is writing the C++ results of the Ball Detection Algorithm to text files and reading them on the other side with Java, where the Multi-Agent Umpiring System runs. Instead of streaming data

to a file, a bridge was implemented by a socket program named PIPE as it can perform faster. PIPE is an inter-process communication mechanism which connects the output of one program to the input of another program without any temporary file. It creates a thread which has a two-way communication link between two programs running on either the same computer or the network. It has a structure in memory that holds the data that is written until it is read. This solution was adopted for bidirectional inter-process communication between these two essential developments and combined them into one system. The architecture of the system is as shown in figure 3.18.

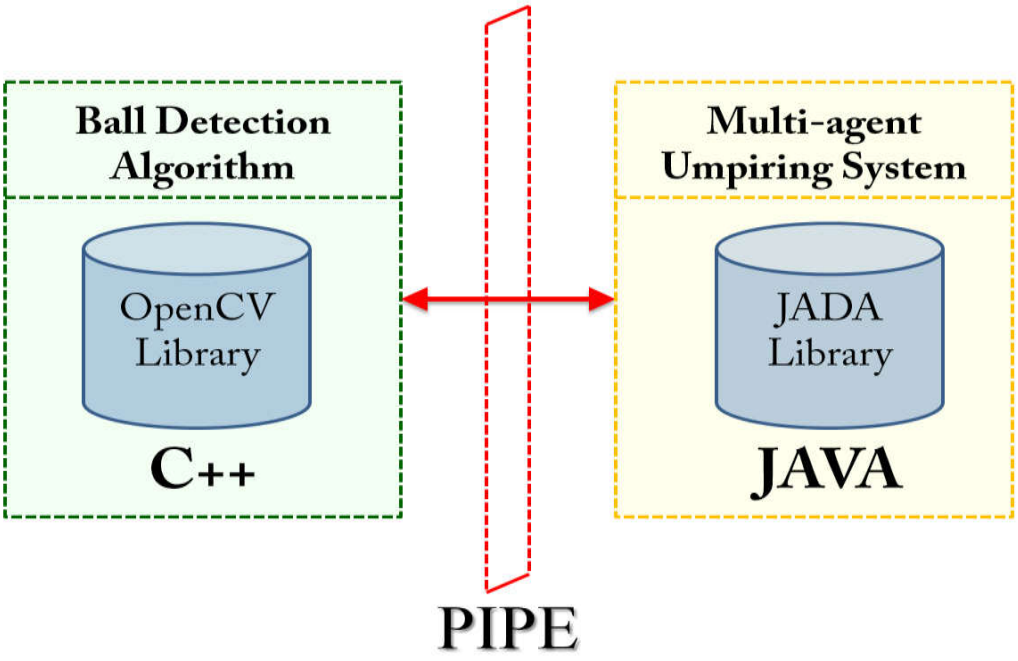


Figure 3.18: Research Implementation (System Architecture)

To validate this PIPE solution works correctly, the ball detection results were printed at the C++ command-line interface before sending them out and counterchecking with the received data at the Multi-Agent Umpiring System's Java Console.

3.5 Test Sequences

To examine the capability of the ball detection algorithm, the system was tested against ten different types of sequences from OUTTDB (Wong and Dooley, 2017). These selected sequences demonstrate the ambiguous and challenging ball detection conditions such as a sudden change of trajectory, occlusion, uneven illumination, multiple object motions, blur and camera noise. Moreover, they were filmed to test the different camera setups.

3.5.1 Single View

Sequence 1

The single-view sequence is extracted from one of the demonstration video files on the web site of the Umpires & Referees Committee of the ITTF (ITTF, 2018). This sequence is composed of 46 frames with (352*240) pixel resolution and a rate of 30 frames per second. The average radius of the image of ball is 4.5 pixels. Because of the low frame rate, object blurring and colour merging with the background can frequently occur. This sequence is selected to demonstrate the detection performance of the system during service as it was is used in Wong and Dooley (2010) for testing their algorithm, enabling a performance comparison. The following figure 3.19 shows one example frame of sequence 1 and a summary of the key features of the tested sequence 1 is shown in Table 3.1.

Figure 3.19: Example Frame in single-view sequence:
A service is about to start

Table 3.1: Summary of the features of Sequence 1

Features of Sequence 1	Single-view sequence
No of frames	46
Identified ball locations	46
Size of frame (pixels)	352×240
Capture rate	30 fps
Average radius of the ball	4.5 pixels
Ball colour	Orange
Key detection challenges	- Low frame rate - Object blurring, merging and occlusion

3.5.2 SBS Stereo-view (Full-Table)

Sequence 2

In this sequence, the stereo cameras were placed further away from the play (approximately 4 meters), to capture the full table view as shown in figure 3.20. Although it can achieve the whole table wider view, the image size of the ball in the second sequence appears much smaller than the other nine sequences, and blurry. This increases the detection challenge. The average radius of the image of ball appears as only 1.95 pixels while the frame width and height are (512*384) pixel respectively. Since the background contains many white horizontal lines as well as ball-like objects, this creates challenging detection scenarios including colour merging and shape confusion. The following figures 3.20 and 3.21 show the capturing arrangement and an example frame of a SBS Stereo-view (Full-table) sequence. A summary of the key features of the tested sequences 2 is shown in Table 3.2 below.

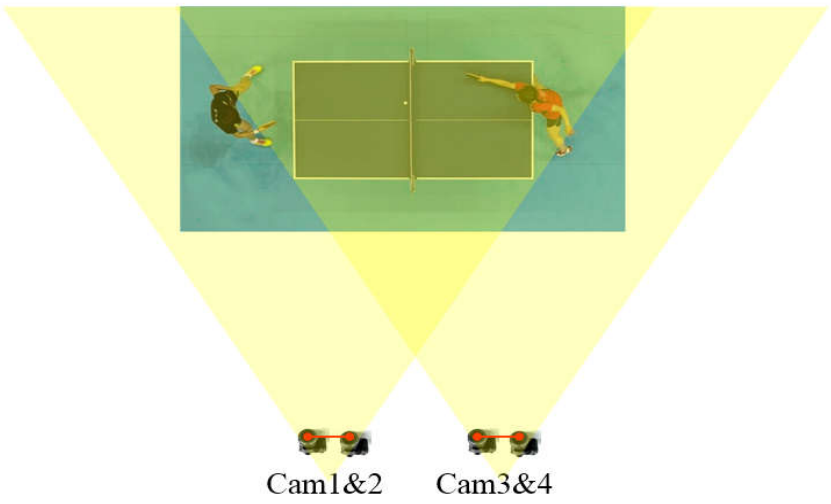


Figure 3.20: SBS Stereo camera arrangement

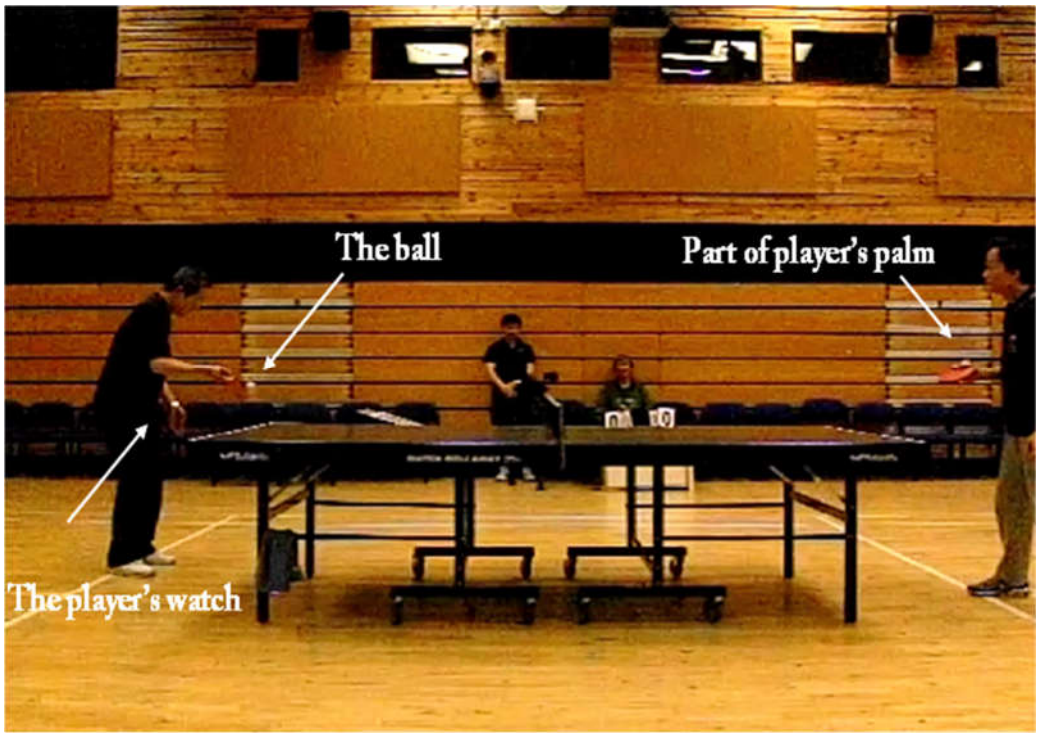


Figure 3.21: SBS Stereo-view (Full-table) sequence

Table 3.2: Summary of the features of Sequence 2

Features of Sequence 2	Stereo view (Full-table)
No of frames	200
Identified ball locations	400
Size of frame (pixels)	512×384
Capture rate	300 fps
Average radius of the ball	1.9 pixels
Ball colour	White
Key detection challenges	- Colour merging, - Object reflection, - Multiple moving objects, - Occlusion, - Complex background, - Very small ball's size

3.5.3 SBS Stereo-view Sequence (Half-Table)

Sequence 3

This sequence is composed of 200 frames with (512*384) pixel resolution and a rate of 300 frames per second. Due to the low resolution of the camera, the sequences appeared darker than the actual match scenery. To get clearer and bigger view of the play, the cameras were placed at a closer location to the table (approximately 2 meters) as shown in Figure 3.22. However, this sequence is selected to test if the developed algorithm is robust enough to detect the ball among confusing objects including ball like images (Score Card), moving parts of referee's body and multiple light illuminations. The following figures 3.22 and 3.23 show the capturing arrangement and an example frame of SBS Stereo-view (Half-table) sequence. A summary of the key features of the tested sequences 3 is shown in Table 3.3 below.

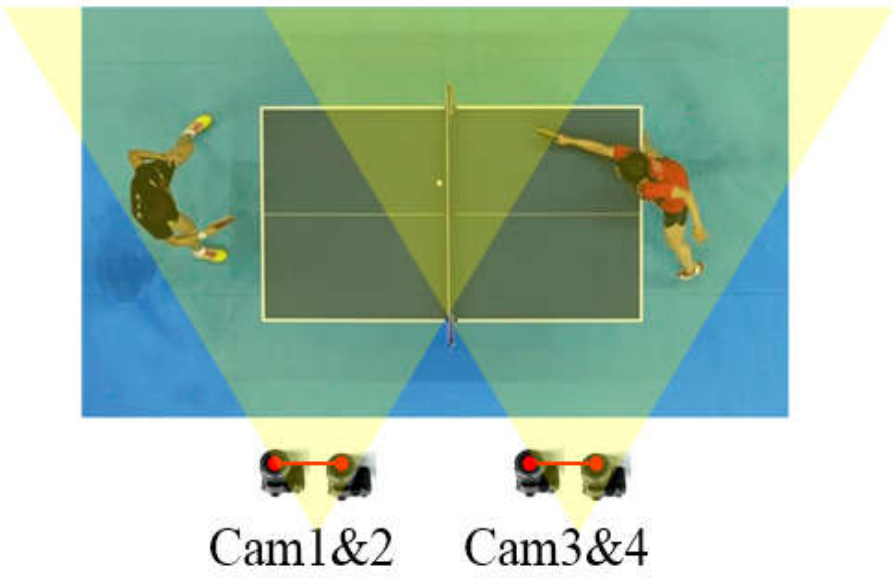


Figure 3.22: SBS camera arrangement

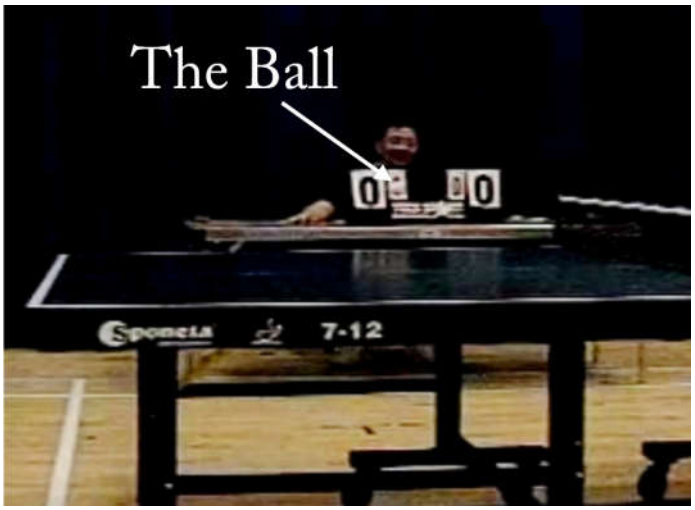


Figure 3.23 Example Frame of the ball crosses the scorecard (Left View)

Table 3.3: Summary of the features of Sequence 3

Features of Sequence 3	Stereo view (Half-table) sequence
No of frames	200
Identified ball locations	400
Size of frame (pixels)	512×384
Capture rate	300 fps
Average radius of the ball	3.4 pixels
Ball colour	White
Key detection challenges	Colour merging, illumination, shape distortion, object reflection, multiple moving objects, occlusion

3.5.4 FTF Multi-View Sequences (Half-Table)

The following multi-view sequences (Sequence 4 to 10) were filmed with the proposed FTF stereo camera arrangement as shown in figure 3.24. These sequences are comprised of different conditions of complete table tennis rallies. Those sequences were used as test sequences in Chapter (5) and (6) in which the different states of each rally are identified. Each view of the sequences has a resolution of 512×384 pixels and was captured at 300 frames per second (fps). The average radius of the image of ball is around 3 pixels. Instead of setting up the camera further away from the play (approximately 4 meters) to capture the full table view, the cameras were placed at a closer location with the table (approximately 2 meters) as shown in Figure 3.24.

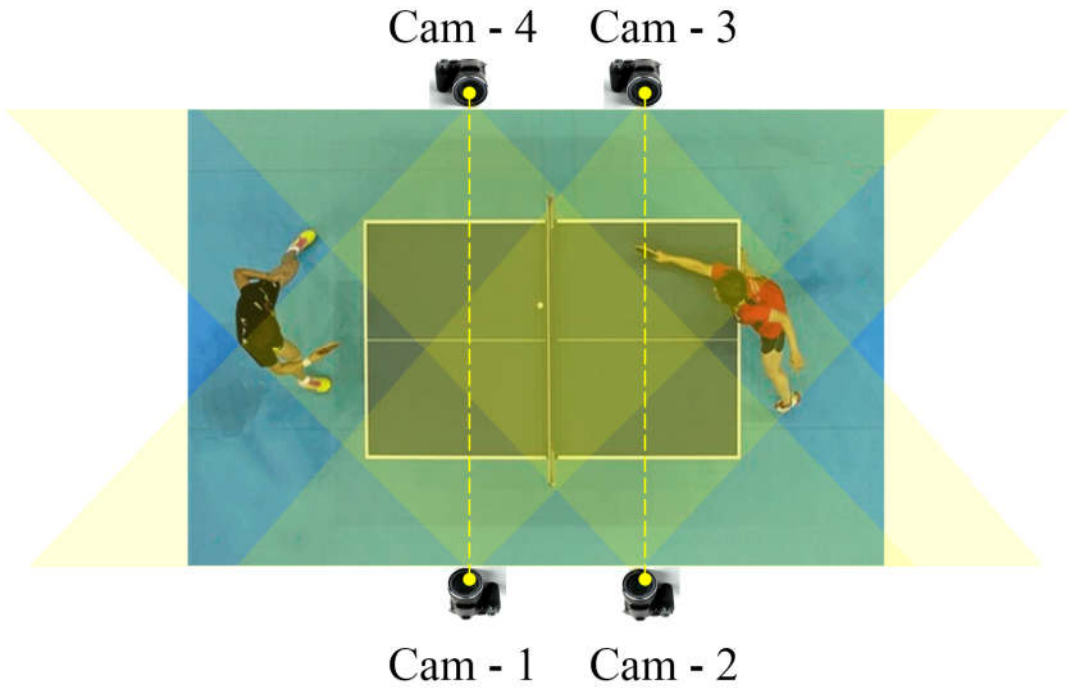


Figure 3.24: FTF Stereo camera arrangement

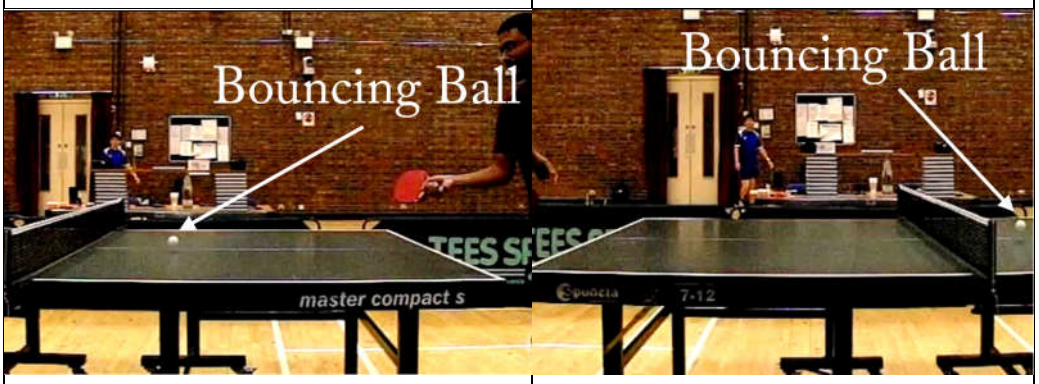
Sequence 4

This sequence is composed of 900 frames and takes 3 seconds to complete a rally. Therefore, a total of 3600 ball locations (on each frame of each view for four cameras) is supposed to identify by the system and compared with the ground truth. As the rally is ended by hitting the net and multiple bounces on the opponent's court, this sequence involves ball detection within multiple motion due to the net vibration. This sequence is selected to demonstrate the detection performance of the system during illumination and uneven lighting conditions. The following figure 3.25 shows an example frame of sequence 4 in each view for four cameras. A summary of the key features of the tested multi-view sequences are shown in Table 3.4 below.

Table 3.4: Summary of the features of Sequence 4

Features of Sequence 4	Stereo view (Full-table)
No of frames	900
Identified ball locations	3156
Size of frame (pixels)	512×384
Capture rate	300 fps
Average radius of the ball	3.4 pixels
Ball colour	White
Key detection challenges	- Uneven illumination and light reflection - Hits the net, Cross over the net - Fault: double bounces on table

Camera 1 View at frame 750 | Camera 2 View at frame 750

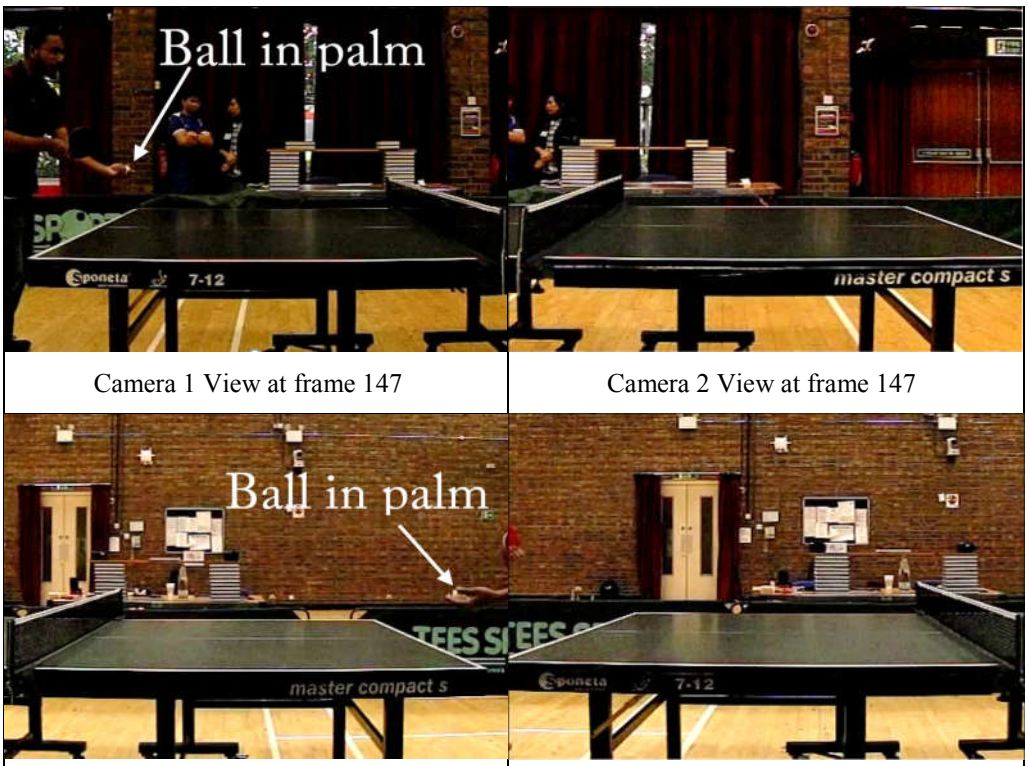


Camera four views at frame 750 | Camera 3 View at frame 750

Figure 3.25: Example Frame: The ball is about to bounce on table

Sequence 5

This sequence is selected to demonstrate the detection performance of the system during service in which several occlusions occur when the ball is blocked by the player's hand and his bat. This is a challenging sequence because the system needs to detect the ball among surrounding objects which exhibit different motions. The following figure 3.26 shows an example frame of sequence 5 in each view for four cameras. A summary of the key features of the tested multi-view sequences is shown in Table 3.5 below.



Camera 1 View at frame 147 | Camera 2 View at frame 147

Camera four views at frame 147 | Camera 3 View at frame 147

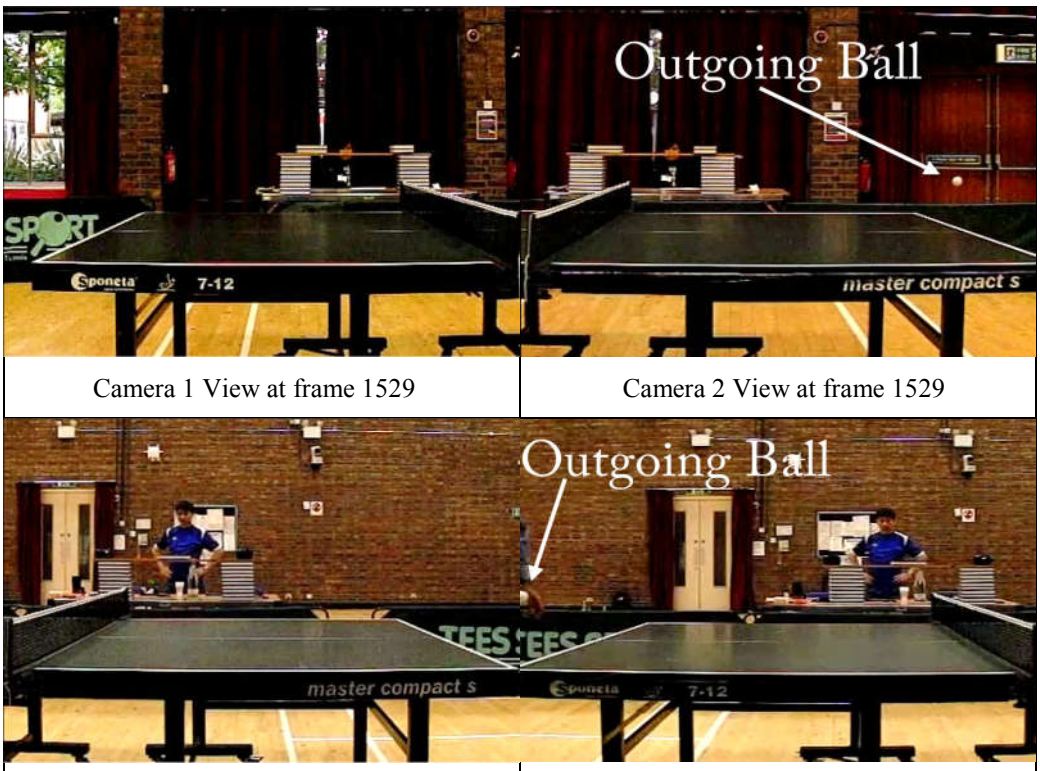
Figure 3.26: Example Frame: The server is about to serve the ball

Table 3.5: Summary of the features of Sequence 5

Features of Sequence 5	Multi-view (Half-table) sequence
No of frames	900
Identified ball locations	3282
Size of frame (pixels)	512×384
Capture rate	300 fps
Average radius of the ball	3.4 pixels
Ball colour	White
Key detection challenges	- Partially see the service (From only Camera 1) - Multiple motion - Ball hits the net - Cross over the net - Disappearance from cameras up to 70 frames - Dropped under table - Fault: ball not received

Sequence 6

Among ten sequences, this is the longest rally and is composed of 1800 frames. Although it is supposed to have 7200 ball locations (on each frame of each view for four cameras), only 6780 ball locations were identified by the system due to the ball reaching out of view or being invisible in cameras' views. In the middle of the play (starting from frame 1570 till 1680), the ball reached out of all the four cameras' views and disappeared for more than 100 frames. This is a challenging sequence because the system needs to be aware that the ball is temporarily out of all cameras' scope and get ready to detect the returning ball. Since the rally is long, it challenges the system's ability in handling and synchronising the ball's position among different cameras. This sequence is selected to demonstrate the multi-view correction and prediction performance of the system during the ball disappearance from all cameras. The following figure 3.27 shows an example frame of sequence 6 in each view. A summary of the key features of the tested multi-view sequences are shown in Table 3.6 below.



Camera 1 View at frame 1529	Camera 2 View at frame 1529
Camera four views at frame 1529	Camera 3 View at frame 1529

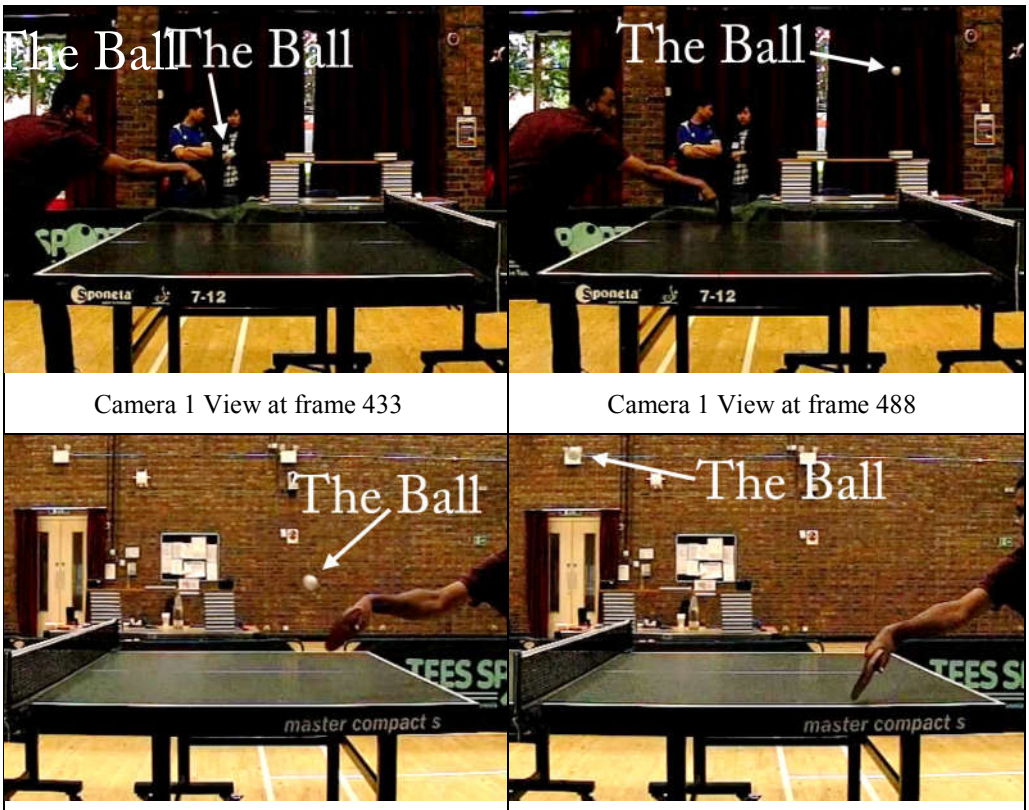
Figure 3.27: Example Frame: Ball about to disappear from all camera views

Table 3.6: Summary of the features of Sequence 6

Features of Sequence 6	Stereo view (Half-table) sequence
No of frames	1800
Identified ball locations	6780
Size of frame (pixels)	512×384
Capture rate	300 fps
Average radius of the ball	3.4 pixels
Ball colour	White
Key detection challenges	- Complex background - Long rally - Multiple Occlusion - Fault: double bounces

Sequence 7

In this sequence, the player is playing the ball in a diagonal direction and striking the ball with force. As the result, the ball is travelling with high speed motion and the apparent sizes of the ball in opposite cameras varies. This sequence is selected to demonstrate the detection performance of the ball's shape variation, dynamic appearance changes and colour merging with background objects. Instead of bouncing at the receiver side, this rally is ended by the ball going beyond the table edge line without bouncing in the opponent's court after being struck by the opponent. The following figure 3.28 shows an example frame of sequence 7 in each view for four cameras. A summary of the key features of the tested multi-view sequences is shown in Table 3.7 below.



Camera 1 View at frame 433	Camera 1 View at frame 488
Camera four views at frame 433	Camera four views at frame 488

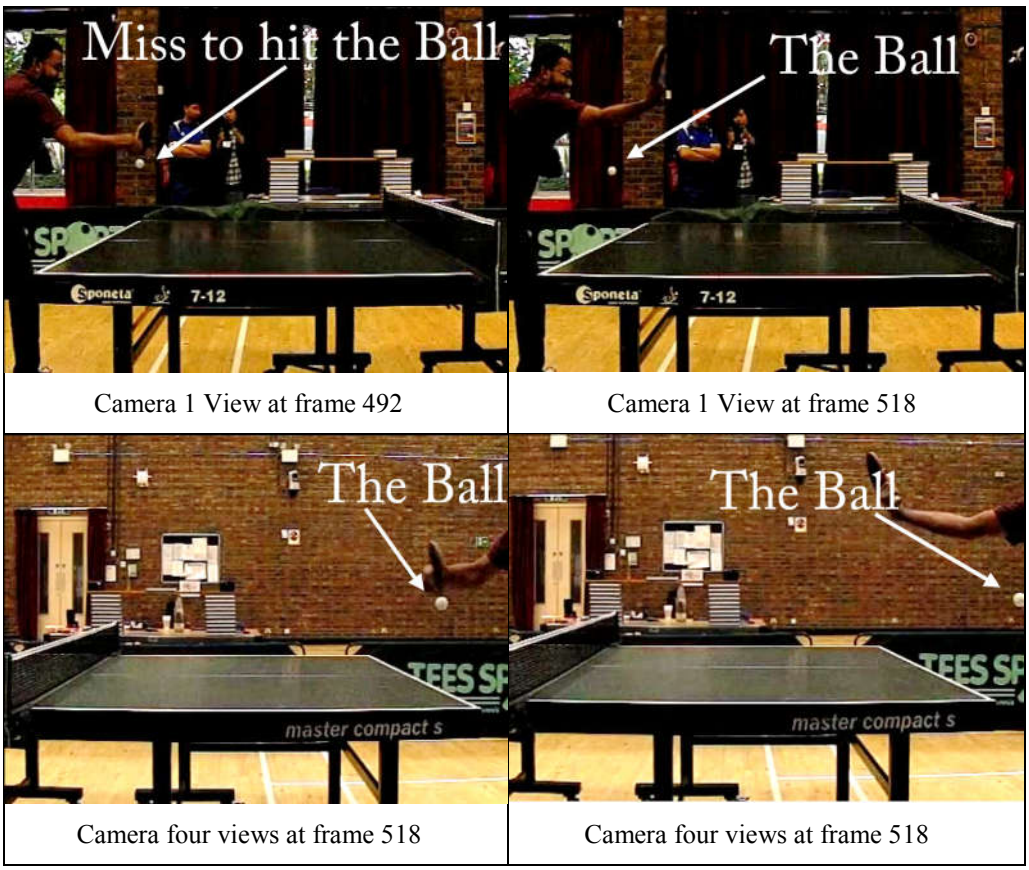
Figure 3.28: Example Frame: Ball about to bounce on table

Table 3.7: Summary of the features of Sequence 7

Features of Sequence 7	Stereo view (Half-table) sequence
No of frames	500
Identified ball locations	1530
Size of frame (pixels)	512×384
Capture rate	300 fps
Average radius of the ball	3.4 pixels
Ball colour	White
Key detection challenges	- Colour merging - Without bouncing at the receiver side - Goes over the table end line - Fault: Out

Sequence 8

The system was intended to test for identifying different types of faults such as faults due to multiple bounces, faults due to a return not bouncing on the right side of the table, and faults due to the ball hitting the floor. In this sequence, the receiver misses the ball after being struck by an opponent. The ball passes over beyond the player end line and it drops under the table. The following figure 3.29 shows an example frame of sequence 8 in each view of cameras 1 and 4. The ball is out of view in cameras 2 and 3. A summary of the key features of the tested multi-view sequences are shown in Table 3.8 below.



Camera 1 View at frame 492	Camera 1 View at frame 518
Camera four views at frame 518	Camera four views at frame 518

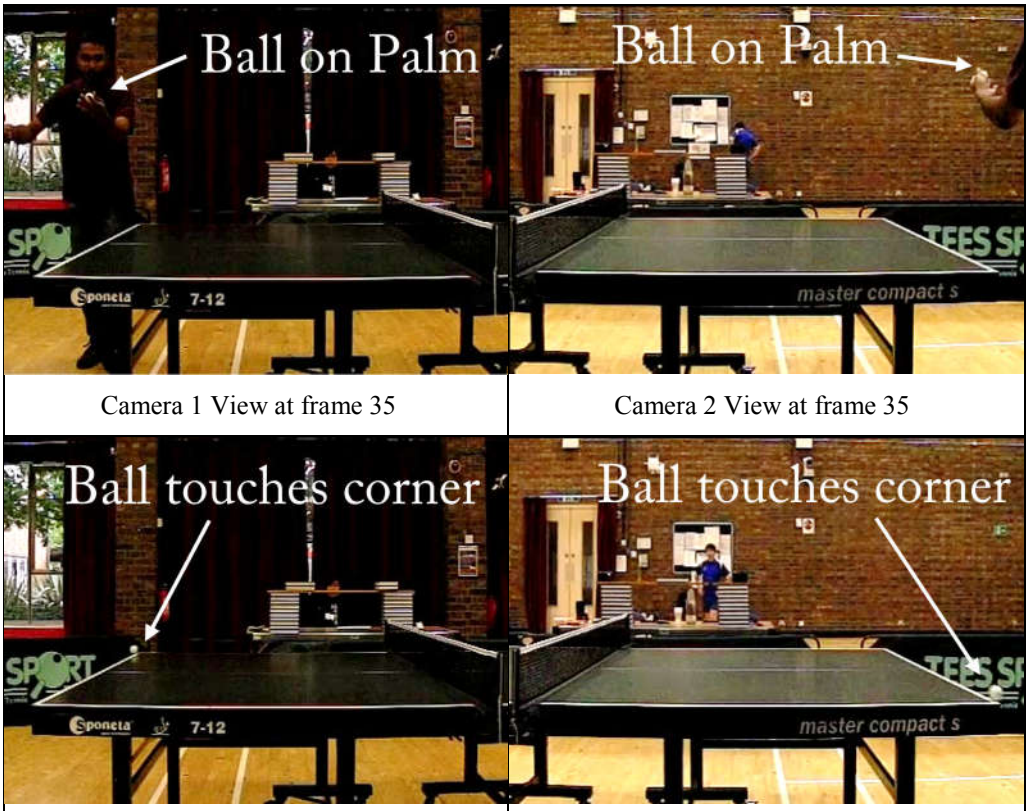
Figure 3.29 Example Frame: Server misses the ball

Table 3.8 Summary of the features of Sequence 8

Features of Sequence 8	Stereo view (Half-table) sequence
No of frames	300
Identified ball locations	722
Size of frame (pixels)	512×384
Capture rate	300 fps
Average radius of the ball	3.4 pixels
Ball colour	White
Key detection challenges	- Uneven lighting - Receiver can not Hit - Miss the ball - Fault: not returning the ball

Sequence 9

In this sequence, the ball progresses from being served until the rally is ended by the ball touching the corner of the table and dropping down to the floor. This also a challenging sequence because to be able to detect whether the ball touches the corner of the table or not, the system needs a robust ball detection algorithm. This sequence is selected to test whether the system can detect the ball location accurately or not and can identify the different type of mistakes. The following figure 3.30 shows an example frame of sequence 9 in each view for four cameras. A summary of the key features of the tested multi-view sequences is shown in Table 3.9 below.



Camera 1 View at frame 35	Camera 2 View at frame 35
Camera four views at frame 569	Camera 3 View at frame 569

Figure 3.30: Example Frame: Ball touches the corner of table

Table 3.9: Summary of the features of Sequence 9

Features of Sequence 9	Stereo view (Half-table) sequence
No of frames	600
Identified ball locations	1988
Size of frame (pixels)	512×384
Capture rate	300 fps
Average radius of the ball	3.4 pixels
Ball colour	White
Key detection challenges	- Shape distortion - Partially see the service and multiple motion - The ball touches the edge of the table - Dropped under table - Fault: not returning the ball

Sequence 10

Among the ten sequences, this is the brightest one with a lot of light reflections. Since all windows are wide open, it has multiple moving background objects which can be confused and indistinguishable from the ball. In this sequence, the ball strongly hits with the net while it is crossing the receiver court to the server and the net gets several vibrations. This sequence is selected to demonstrate the detection performance of the system during multiple motion allied with object blurring and colour deviation impact due to the uneven lighting of the scene. The following figure 3.31 shows an example frame of sequence 10 in each view for four cameras in which the ball is about to get struck by server. In this particular frame, only one camera can see the ball while it is out of view from the other cameras. A summary of the key features of the tested multi-view sequences is shown in Table 3.10 below.

Camera 1 View at frame 342 | Camera 2 View at frame 342

Camera four views at frame 342 | Camera 3 View at frame 342

Figure 3.31: Example Frame: Ball is about to get strike by server

Table 3.10: Summary of the features of Sequence 10

Features of Sequence 10	Stereo view (Half-table) sequence
No of frames	1200
Identified ball locations	3740
Size of frame (pixels)	512×384
Capture rate	300 fps
Average radius of the ball	3.4 pixels
Ball colour	White
Key detection challenges	- Very bright and the most complicated background - Multiple moving objects - The ball hits the net - Crosses over the net - Fault: double bounce

3.6 Evaluation Methodology

To appraise the performance of any system, it is very important to choose the most appropriate and consistent assessment metrics. While there is a publication related with umpiring the service part of a table tennis rally (Wong and Dooley, 2010) comparing the result for single-view video, there are no stereo datasets available as a ground truth to compare the result. For that reason, a sensible way to evaluate the performance of the system is to compare the ball locations identified by an ordinary human (umpire) and the ball detection system. To create such a comparator (loosely referred to as the ground truth in this thesis), human volunteers were invited to identify the ball locations on each frame of each view. Those results have been maintained at the OUTTD (Wong and Dooley, 2017) and shared with research community for non-commercial research purposes.

3.6.1 Performance Metrics

In statistics, the Root Mean Square Error (RMSE) is the standard deviation of the average of the squares of the errors that is, the difference between the estimator and what is estimated (Hyndman and Koehler, 2006). Throughout this thesis, the RMSE between the actual location of the ball and system detected location would naturally be considered as a measure of success. It is consistently applied as a comparator in critically evaluating the performance the performance of the system against the ground truth and the suite of developed algorithms in subsequent chapters. As it is targeted to be an umpiring system, it is necessary to achieve the high detection rate and low processing time. As a result, three key

quantitative parameters: the RMSE, detection rate and processing time have been used in this thesis as benchmarks to validate all the performance results.

3.6.2 Software Code Validation

When developing the software system, the code verification activity requires checking the code against human error. Throughout the implementation of the system, various code verification and validation techniques (Margaria and Steffen, 2016; Wallace et al., 1996) are used to verify correct implementation of software design into code. To evaluate the source code for accuracy, correctness and testability, the code verification involves following tasks;

- **Cppcheck** (Cppcheck - A tool for static C/C++ code analysis, n.d.) is used for detecting errors (static analysis checks) in the code. It provides unique code analysis to detect bugs and focuses on detecting undefined behaviour and dangerous coding constructs.
- Breaking down the program and independently testing the functionality of each component and comparing the result with Visual Basic Editor and Microsoft Excel.
- Evaluating source code for compliance with code standards and language standards.
- For the Knowledge-Based System (KBS) development, conducting a logical versification of the structure of the knowledge and rules in the knowledge base for consistency, completeness, etc.

3.6.3 Quality Assessment Methods

The system is tested against 10 table tennis match sequences which contain different events of a typical table tennis rally and can confidently be used to test the system. The ground truth's trajectory is based on the set of ball locations identified by human volunteers on each frame of each view. If the system provides the trajectories that are well aligned with the ground truth's, it indicates that the detection is successful throughout the whole sequence and acceptable for umpiring.

Likewise, if the system can detect the ball with average RMSE less than the radius of the ball, the detection rate is acceptable for developing an umpiring system. While the detection rate is a concern, timing is also another consideration factor for developing an umpiring system. In computer science, a real-time system means the programs must produce a result within specified time constraints. The maximum acceptable delay might be somewhere between 3 and 5 sec but different systems can have a range of acceptable durations to produce the results, and the acceptable delay depends on operational studies that assess human reactions to the system (Ben-Ari, 2006). In this research, the system is designed with the acceptable delay of 0.028 sec per frame and 8.4 sec for processing 300 frames. If each agent of the MAS is run on a separate computer, this time is expected to be reduced to comparable time delay with human umpire. Detail result discussion can be found in Chapter 5, Section 5.6.

3.7 Summary

This chapter describes the system development processes which are conducted in this thesis. The research methodology, testbed development, platform selection and tools used, and the experimental setup have been presented in this chapter. Finally, a brief discussion on the evaluation methodology together with the key performance indicators, validation procedure and quality assessment methods have been provided. The new ball detection algorithm developed will be presented in Chapter 4.

Chapter 4

Detecting a Table Tennis Ball for Umpiring

(Contribution Chapter)

4.1 Introduction

This chapter presents an implementation of block 4 (Ball Detection), part of the proposed framework which has presented in Chapter 1, figure 1.2. As mentioned in Chapter 1 and 2, tracking the location of a table tennis ball during a match for umpiring purposes is a challenging task for a number of reasons including:

- the small size of the ball (40 mm in diameter)
- the ball often travels at high speed (around 112.5 kilometres per hour)
- multiple moving objects nearby (bat and players)
- complicated backgrounds (advertising logos, audience)
- uneven lighting
- occlusion
- the time constraint
- the computational cost
- the installation cost and etc.

Since the ball often travels in high speed, the images of the ball are changing as it moves from one frame to another through the field of view of a camera. As a result, features of the ball such as colour, shape and size get distorted in captured image, and it may be deformed or blurred. To solve these problems, identifying the ball considering different features together with recovery mechanisms could overcome various detecting challenges and complement the weaknesses of each. However, a careful consideration of selecting the right method is important since too many combinations of different methods will slow down the system. While the overall process needs to be conducted in real-time, high detection rate also plays a crucial role in decision making, and wrong detections can seriously impact the automatic umpiring system.

According to the survey in object detection and tracking literature in Chapter 2, feature based detection is the common methods in several publications. Even though they have achieved significant experimental results, many of them (Liu et al., 2014; Bao et al., 2012; Chen et al., 2010, 2011; Zhang et al., 2010; Arenas et al., 2009) have tested in ideal situations such as a laboratory environment for virtual gaming purposes rather than an actual match scene. In a laboratory environment, it is straightforward to identify the feature of the ball against a plain background (Fitriana et al., 2016; Qazi et al., 2015; Bao et al., 2012; Chen et al., 2010, 2011; Arenas et al., 2009). However, detection becomes far more challenging when there is for example, another object with similar features adjacent to the target ball, while background clutter such as audience or advertising boards make consistent ball detection extremely difficult in some real match situations. Moreover, due to changes of illumination in the scene, there is

no guarantee that the feature will be same for the same ball in all frames. This leads to inaccurate detection with feature-based detection and tracking.

While the proprietary ball detection systems such as **Hawk-eye, Goal-line and GoalRef®** (Wei et al., 2016; Bal and Dureja, 2012; Dovaston and Correspondent, 2012; Owens et al., 2003) are applied in real matches, they have to employ multiple broadcast-grade high speed cameras and fit them at high locations to provide aerial views of the ball against a simple background of the court. The proposed automatic umpiring system is aimed to be portable and accessible to amateur users. The Hawk-eye system is fixed and expensive and is therefore not suitable.

An alternative method is object detection and tracking based on its motion. As discussed in Chapter 2: Section 2.2, among several techniques in motion detection (Abdelli and Choi, 2017; Xu et al., 2017; Chakroun et al., 2011; Tong-yao, 2011, 2011; Zivkovic, 2004; Toyama et al., 1999), BS by frame differencing is widely applied with low complexity. While this method is simple and effective, it will only work if the surrounding area of the ball remains unchanged and the object is in motion across consecutive frames. Motion detection cannot detect stationary objects. Therefore, additional methods needed to be applied in order to detect a stopped ball. Moreover, not only the ball but also players and table tennis bats are moving with different motions in real table tennis match sequences. Therefore, ball detection purely based on motion is also prone to errors.

As mentioned, the strength and weakness of different tracking algorithms in Chapter 2: Section 2.2.3, tracking is based on series of detection. While the **KF**

(R. E. Kalman 2001), **Extended KF** (Yilmaz et al., 2006), **Mean Shift Tracker** (Comaniciu and Meer, 2002), **Particle filter** (Wang et al., 2016) and **Support Vector Machine** (Zheng et al., 2012; Huang et al., 2006) are commonly used in tracking, most of these trackers perform as iterative methods by recursively updating an estimate location based on a series of measurements observed over time. This require high computation, time consumption and are not suitable for real-time tracking. On the other hand, **Motion models** have been widely used in ball tracking (Maksai et al., 2016; Seo and Wuest, 2016; Takahashi et al., 2015) because of their simplicity and low computation. While the motion model is generally reliable, the predicted location can be erratic if the detected location in the previous frame(s) is wrong. Although the predicted ball position can be used to recover for a few undetected frames, the tracking itself can get lost if detection is missed in several consecutive frames. To prevent this error from propagating, a new mechanism is required to evaluate and reset the centre of the predicted location whenever necessary.

Another major challenge of detection and tracking is partial or full occlusion, which happens quite often in table tennis match sequences when the image of the ball is blocked by the players or their bats. To address this problem, multiple cameras approach has been incorporated for ball detection as explained in Chapter 2: Section 2.5.2. Although the chance of capturing the ball is higher with **multi-view approach** (Anuj and Krishna, 2017; Cheng et al., 2016; Takahashi et al., 2016; Liu et al., 2014; Bal and Dureja, 2012; Bao et al., 2012; Chen et al., 2010, 2011; Zhang et al., 2010; Arenas et al., 2009; Owens et al., 2003), it increases the processing time and requires a high degree of co-ordination

between different views as well as an ability to resolve conflicts when inconsistent information is acquired.

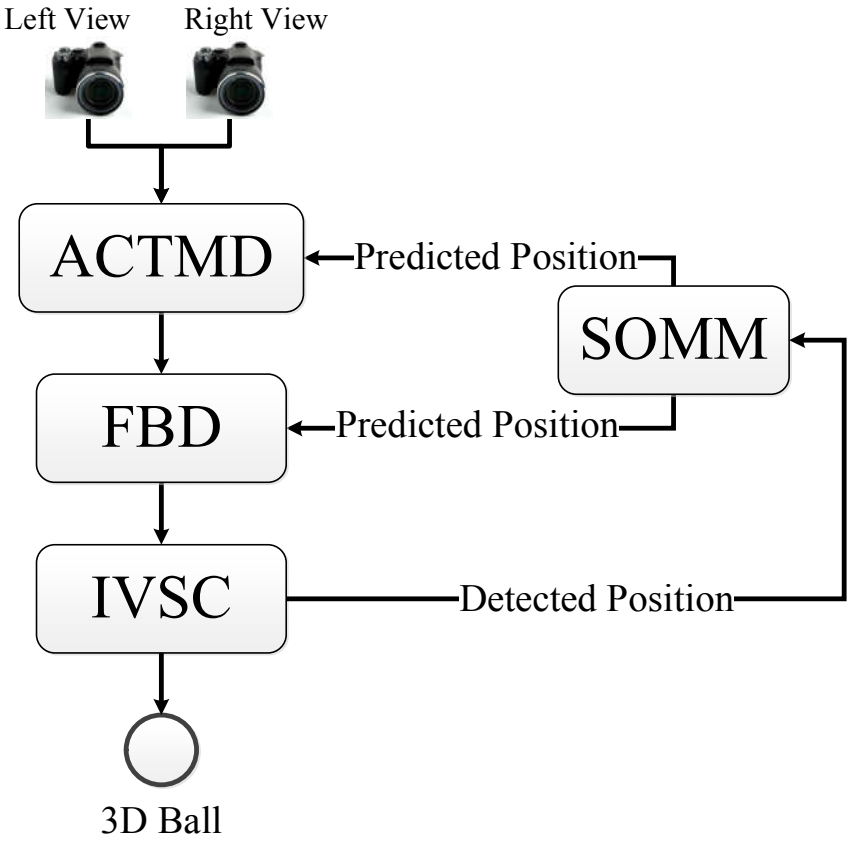
To summarise, this chapter presents a multi-view detection and tracking strategy that addresses the problems described above and can track a table tennis ball efficiently. Like various approaches discussed in Chapter 2, the proposed system also follows the segmentation-detection-tracking of sequences, but emphasis on segmentation enhancement, develop a new inter-view correction technique for detection improvements and a mechanism for evaluating the tracking. The basic framework for the testbed has been developed for experimenting different ideas and evaluating the algorithm. The performance of the algorithm is critically evaluated with real match scenes sequences from the **OUTTD** (Wong and Dooley, 2017) as described in Section 3.4. Experimental results show that the developed algorithm is robust enough to distinguish the ball from a complex dynamic background and provides satisfactory precision in detecting and tracking. The remainder of the chapter is organised as follows: Section 2 presents the proposed detection and tracking strategy while Section 3 briefly discusses the experimental setup and the chosen tested sequences. The outcome of the results and performance comparison are given to verify the effectiveness of the proposed algorithms in Section 4. Finally, a conclusion is provided in Section 5.

4.2 Ball Detection and Tracking Algorithm

While the ultimate goal is to build an automatic umpiring system, one important activity of such a system is to accurately and rapidly track the location of the ball during a match. For an affordable automatic umpiring system intended for wide use, a stereo system formed by pairing two single-view cameras was identified in previous chapters as being appropriate. This research exploits the best use of statistical analysis of visual features together with temporal analysis of motion information and develops a hybrid segmentation method which is named as **Adaptive Colour Thresholding and Motion Detection (ACTMD)** module for ball segmentation. A **Second Order Motion Model (SOMM)** module is implemented for predicting and tracking the travelling direction of the ball. It conducts by taking some measurements of the ball such as position, velocity, and acceleration. The SOMM module provides a guidance for both of detection and segmentation. Hence, the **Feature based Ball Detection (FBD)** is used for identifying the ball based on its features and evaluating with the closest location with the predicted ball's position. To address the problem of occlusion, multiple cameras have been incorporated in the algorithm and a new **Inter-View-Self-Correction (IVSC)** module, which uses a positional information from another view to estimate the location of the ball in current view, is proposed. An additional benefit is that the depth information can be derived from two overlapping views and that information can be used to predict the 3D trajectory based on a triangulation theory which can be seen in *Section 3.4.3*.

To summarise, Figure 4.1 illustrates the block diagram of the proposed system which consists of four main modules:

- ACTMD module for segmentation,
- SOMM module for trajectory prediction,
- FBD module and
- IVSC module.



ACTMD: Adaptive Colour Thresholding and Motion Detection
SOMM: Second Order Motion Model
FBD: Feature based Ball Detection
IVSC: Inter-View Self-Correction

Figure 4.1: Block diagram of the proposed system

The first step of the detection process is initialising the parameters such as frame rate, width and height, sensor size, camera's position and other essential parameters that will be input to the system. The location of the ball, colour value, diameter and all other necessary parameters are then dynamically updated in subsequent frames. Only the essential region of a frame which contains the ball, known as the region of interest (ROI) is processed to improve the system efficiency. In Figure 4.2, the ROI which is defined as a small rectangle region with the width of the side set to two to three times of the diameter of the ball depending on the detection situation. This ensures that the region is a small portion of the frame yet big enough to fully contain the ball. As the location of the centre of ROI is controlled by the SOMM module, it can dynamically adjust to where the predicted centre of the ball is at each frame. By this way, it can narrow down the search area.

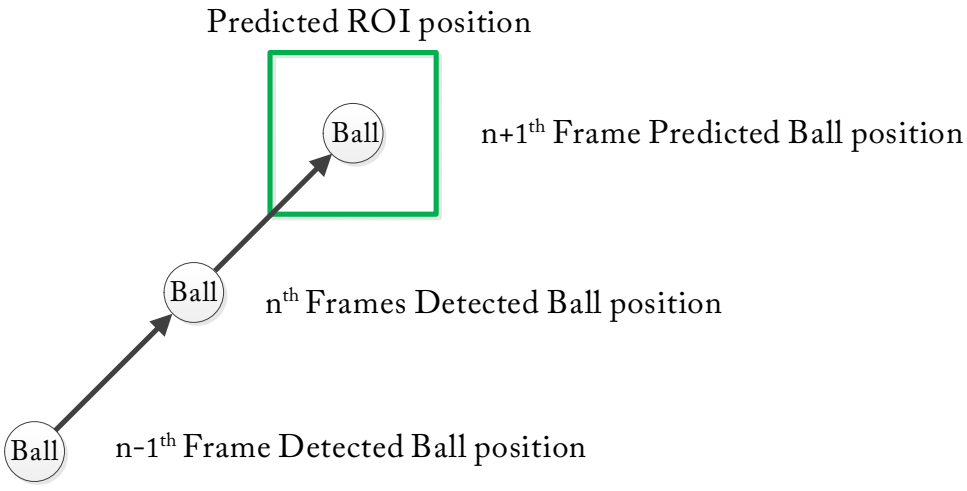


Figure 4.2: Predicted ROI position

If the ball is undetected in a frame, ROI will increase and restore to its default size once the ball is found. If the ball reaches out of the camera's viewing

angle, the ROI will expand its width and height, but the ROI cannot become larger than the frame size. If the ball goes out at the right or left corner, the ROI expands to full frame height as the ball may return from anywhere on that side. It is less likely to return from the other side because of the direction of play. After that, the image enclosed by ROI can then be converted to HSV colour space. In HSV, hue means the wavelength of the colour percept, Saturation stands for the amount of white light present in the colour and Value represents the intensity of the colour which is also known as brightness. By default, standard image capturing devices produce images in RGB colour space which stands for a combination of Red, Green, and Blue colour. To represent light intensity, the incoming input video stream composed of RGB colour is converted to HSV colour space and it is thresholded based on the maximum and minimum allowable colours, called a Binarization Process.

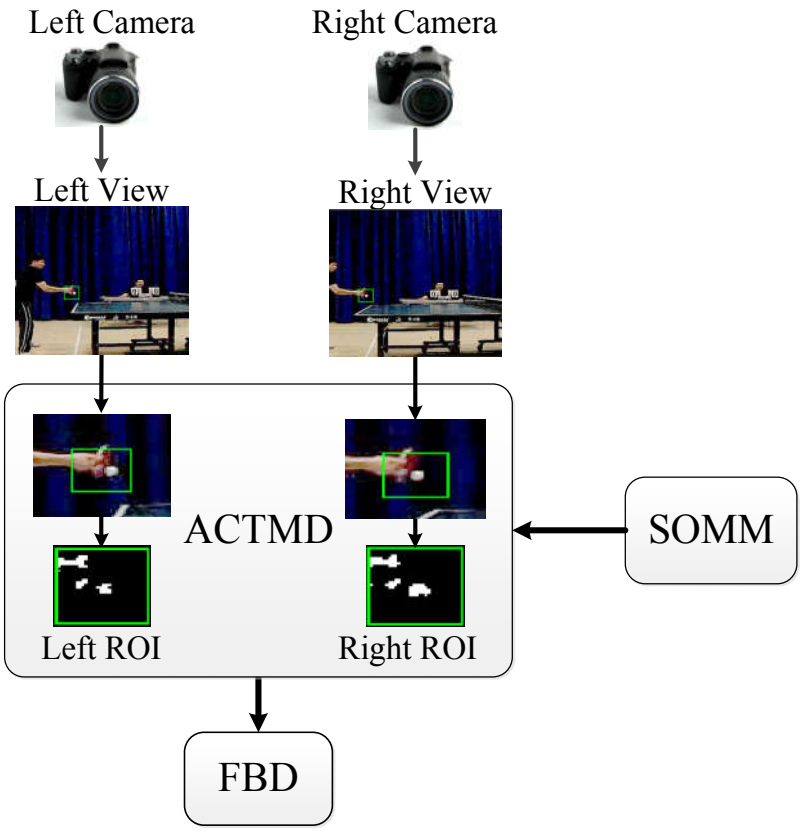
4.2.1 Adaptive Colour Thresholding and Motion Detection (ACTMD)

As mentioned in *Section 4.1*, employing either colour-based thresholding or motion-based segmentation alone is not good enough to reach the expected level of successful detection. The main problem of colour-based-segmentation method is that the colour values of the ball may be vary according to the lighting condition of the environment which requires the operator to manually change the colour range used to segment the ball (Putri et al., 2017). Moreover, the upper portion of the ball tends to be brighter than the lower part as light sources are usually located in the ceiling. When the ball is in high-motion, the image of the

ball is distorted, and its colour becomes blurred. That makes it very difficult to set a proper threshold in CT. Too-loose-threshold may result in too many irrelevant objects left in the binary image (Wong, 2008) which will waste a lot of processing time to analyse, but too-tight-threshold may filter out the right ball as the colour of the ball can be vary along the travelling due to the uneven lighting.

On the other hand, while the adjacent frames differencing technique can roughly distinguish the moving objects from the background, the BS will work only if the background remains unchanged across the consecutive frames. The fluctuating intensity of the ball can be apparent if the video is captured at a rate much higher than the frequency of the electricity supply of the lights (e.g. 50 Hz). Moreover, BS doesn't always provide the right segmentation when the ball is on the player's palm with no motion, or there are multiple moving objects with similar motions. It is therefore desirable to have a combined CT and BS method with its thresholds dynamically updated.

To complement this, the ACTMD module jointly uses the results of CT and BS under demanding situations such as colour matching or no motion. By automatically switching CT or BS method based on the segmented result, ACTMD becomes a very effective module to overcome the above segmentation problems. Figure 4.3 illustrates the block diagram of the proposed ACTMD.



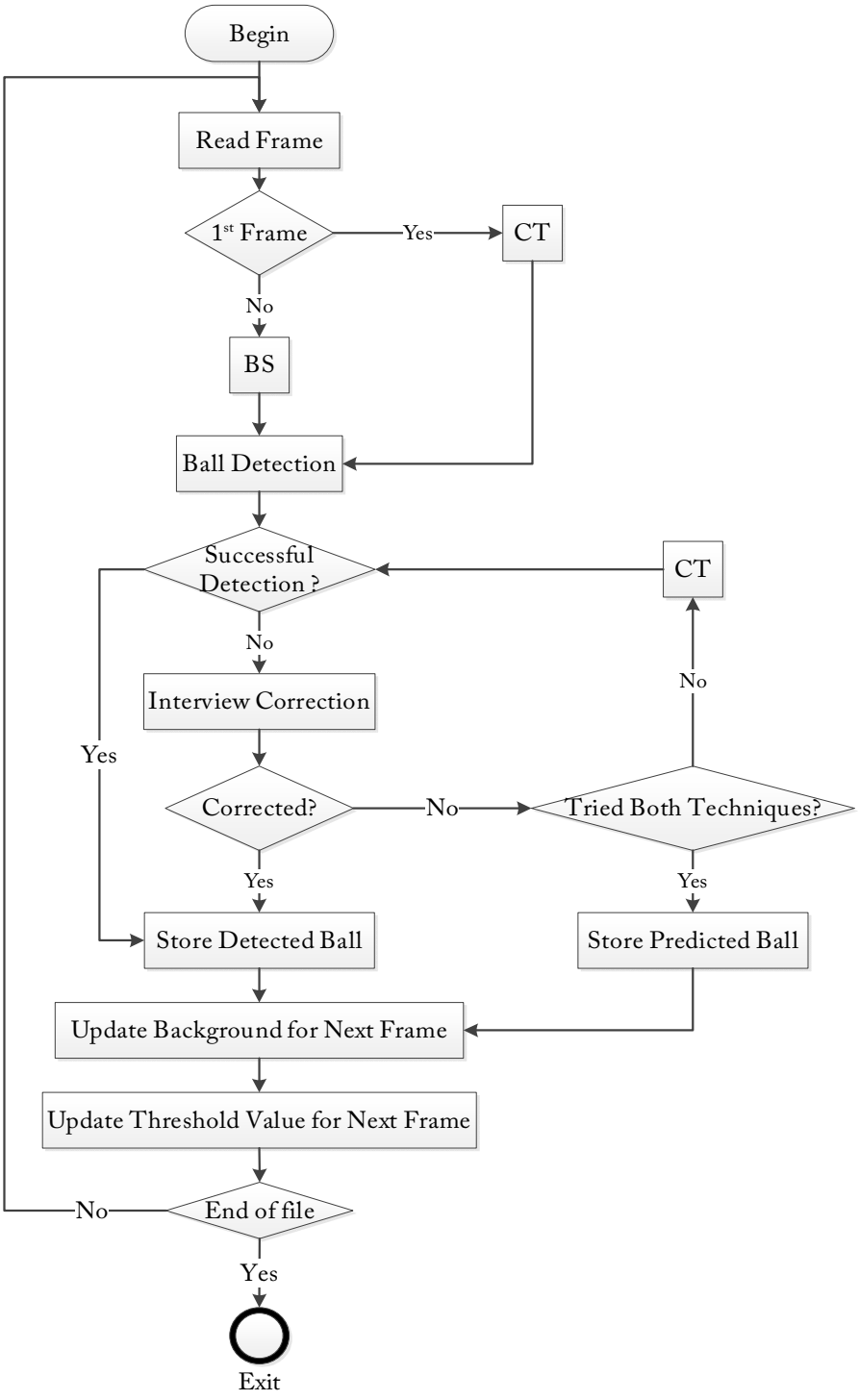
ACTMD: Adaptive Colour Thresholding and Motion Detection

SOMM: Second Order Motion Model

FBD: Feature based Ball Detection

Figure 4.3: Block diagram of the proposed ACTMD

The simplified Flowchart of the developed algorithm is shown below in figure 4.4 in which Detected Ball means the successfully detected ball and Predicted Ball is the predicted location of the ball based on previous detection results.



Where;

CT: Colour Thresholding

BS: Background Subtraction

Figure 4.4: Flowchart of ACTMD module

As the system can only establish the background based on the previous frame, BS is unusable for the first frame. Therefore, ACTMD subtract the ROI by predefined CT range. The predefined threshold value is used only for the first frame thresholding in the beginning. After that, that value will be automatically updated based on the result of detected ball. Once a ball is detected, the system will tune the colour thresholded value for next frame by extracting the pixels of an area at the detected location of the ball in the previous frame. It will calculate the mean and standard deviation of the detected ball's inner HSV pixel values to reject outliers. The acceptable range of pixel values (R_i) is defined using Equation (1):

$$t\ (\mu_i - m_i\sigma_i) < \mathrm{R_i} < t\ (\mu_i + m_i\sigma_i) \quad (1)$$

where: i is the index of the colour channels. μ_i and σ_i are the mean and standard deviation colour values of the inner area of the previous detected ball (PDB). Whenever the system can successfully detect the ball, the μ_i and σ_i will be recalculated and the threshold margin will be dynamically updated to reflect the detected colour of the ball. While the system can automatically and adaptively update the threshold margin, the user can adjust the minimum and maximum threshold margin (whether to be a tight threshold or a relaxed threshold), by assigning the tolerance value (t) and the multiple value (m_i), where t and m are real number between 1 and 2.

As for BS, ACTMD segment the candidate moving ball from the ROI by detecting its motion. The frame differencing is the common method used in BS based on motion. This method adopts pixel-based difference to find the moving

ball. Then, the image is binarised according to the segmented results. In the areas that contain moving objects, the pixels whose grey value is greater than a threshold are set to 1. Otherwise, the pixels are 0, i.e.

$$I_b(x,y) = 1, \ \ if\ I_{fd}(x,y) \geq I_T \tag{2}$$

where: $I_b(x,y)$ is the value of pixel (x, y) in the binary image, and I_T is a threshold for binarisation. $I_{fd}(x,y)$ is the value of pixel (x, y) which is the result of frame differencing. Regarding with background updating, the system was tested to develop a background model by averaging a set of previous frames and kept continuous updating it. While averaging a number of previous frames, a multiple duplication of balls from those frames are appeared as several white circle shadows in segmentation result and it becomes very difficult to select the right one. Moreover, due to the 50 Hz variation of the intensity of the lights, a good background could not be established this way. As a result, only the previous frame is used as the background. This simplifies the task and decreases the computation. However, the drawback of this approach is that when the ball is not traveling very fast, the images of the ball in the background and current frame are partially overlapped. The subtraction of these two images often results in two crescent-shaped objects situated at equal distance from the predicted ball location as shown in Figure 4.5.

Figure 4.5: Two crescent-shaped objects in ROI

To remove the crescent corresponding to the ball image in the background, the direction of ball travel (calculated by SOMM) is used to guide the removal (e.g. if the ball is travelling from left to right, the crescent on the left will be removed).

In that scenario where detection is unsuccessful due to no motion or several motions in the ROI, the segmentation will be attempted again with CT. If the candidate ball is found in the ROI, it will proceed to the detection stage for further identification, and the location of the centre of ROI will be dynamically updated to where the predicted centre of the ball is in the next frame. The predicted ball location is determined by the SOMM module which will be discussed next.

4.2.2 Second Order Motion Model (SOMM)

For tackling the sudden change of trajectory, the video should be filmed at a frame rate that is high enough to capture all the important movements of the ball but low enough to reduce processing time. To be within the capabilities of low-cost cameras, the maximum capture rate used was 300 frames per second

(fps), to capture the fast traveling ball more clearly and with less blurring. By this way, the displacement and velocity of the ball between succession frames become relatively smaller and a simple motion model can be used to model the trajectory. In SOMM module, a SOMM is used for predicting the locations of the next candidate balls, investigating the travelling speed, direction of the ball, and forming a trajectory. As the second order model requires at least two data points to calculate its velocity, the actual prediction will not take place until frame 2. As can be seen in equation (3), the predicted location is calculated by using the detected location for the current frame together with the ball locations from previous frames, so the predicted ball location of the $n+1^{th}$ frame P_{n+1} is given as:

$$P_{n+1} = \begin{cases} B_n & if\ \ n = 1 \\ B_{n-1} + v_n \Delta t & if\ \ n = 2 \\ B_{n-1} + v_n \Delta t + \frac{1}{2} a_n \Delta t^2 & if\ \ n \geq 3 \end{cases} \quad (3)$$

Where: $\Delta t = 1/fps$, Δt is the time difference between the two frames in which the ball is successfully detected and a_n is the ball's acceleration at frame n. C_1 is the centre of the ball in the first frame. B_{n-1} is the detected ball location in the previous frame, v_n is the velocity at frame n. Apart from tracking the centre of the ball, the SOMM also predicts the apparent radius of the ball in the next incoming frame by averaging the results of five previous frames. This information will be useful in umpiring whether the ball touches the table or not. The trajectory of the ball is estimated and predicted only in the frame where the ball was detected successfully with the total detection score higher than two. The assignment of detection score will be explained in coming section 4.2.3.

4.2.3 Feature-based Ball Detection (FBD)

While the motion model is generally reliable, the predicted location can be erratic if the detected location in the previous frame(s) is wrong. To prevent errors from propagating, FBD module identifies another reference point which is called the centre of the object that is **most likely the ball (OMLB)** for predicted ball's position evaluation. To achieve OMLB, FBD retrieves all the contours from the ACTMD's segmented binary image. After that, a morphological opening and closing operation is applied to eliminate holes and very small objects from these contours. It is followed by Gaussian smoothing to reduce noises from the segmented contours. During contour detection, if only one contour is segmented from the binarised ROI, the system will select the centre of that contour as a reference point, OMLB. If more than one contours are segmented, the system will calculate the Euclidean distance between the centre of the predicted ball and the centre of each candidate contour until the end of candidate contours. A segmented object whose contour encloses the centre of the predicted location and with the shortest Euclidean distance between its centre and the predicted location, will be deemed the candidate ball and is assigned as the OMLB. The simplified Flowchart of the process of nearest contour detection to achieve OMLB is shown below in figure 4.6.

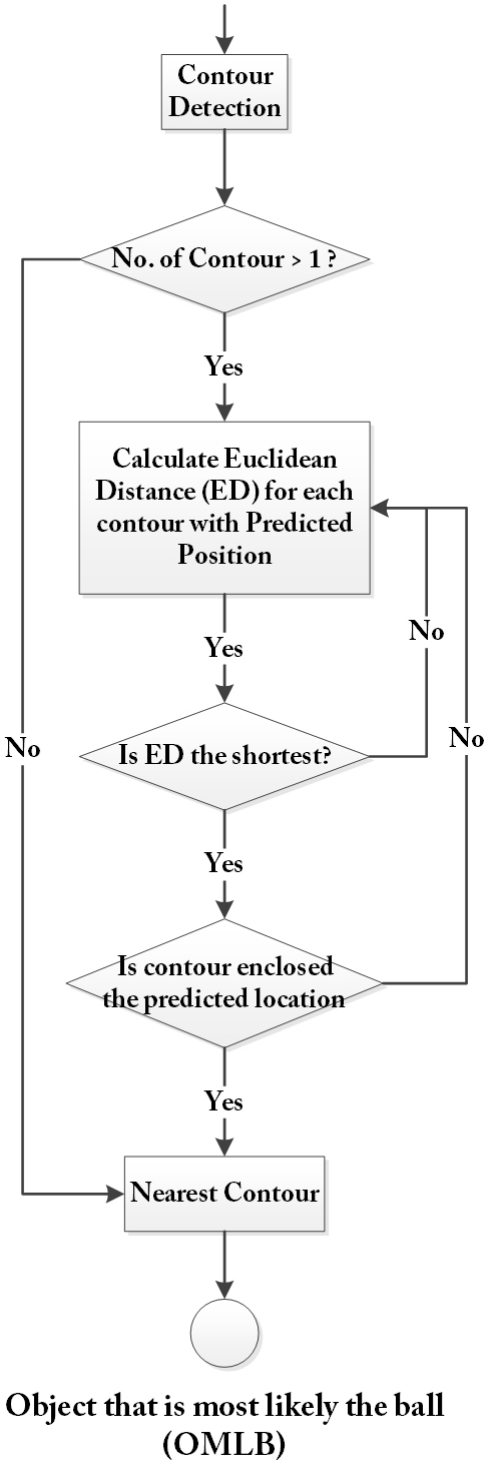


Figure 4.6: Process of nearest contour detection to achieve OMLB

In figure 4.6, if only one contour is segmented from ROI and its distance is significantly different from the predicted location, this may indicate that the predicted ball location has an error. In this scenario, the centre of predicted ball location will be reset to the centre of OMLB. As the contours of these objects are not necessarily circular, the centre of the object is therefore calculated by averaging positions of all the pixels enclosed by its contour.

After identifying the OMLB, the FBD module identifies all segmented objects by comparing the expected features such as size, shape and location. In FBD, the detection procedure is improved by incorporating a variety of features and the characteristics of candidate balls are compared with the actual ball. The one with minimal error is deemed the detected ball. Since the image of a table tennis ball is most likely to be circular, the Hough Circle Transform (Bradski and Kaehler, 2008b) and Canny edge detector (Canny, 1986) along with an object evaluation technique is employed to the smoothed segmented objects for identifying the best candidate ball. In the figure 4.7(a), the red circle is the circumference of the detected ball and the green point defines the centre position of the detected ball. The blue segmented object in figure 4.7(b) represents the predicted ball while, the green circumference is determined by employing nearest contour detection as OMLB. The multiple red circles in 4.7(b) are candidate circles which are detected by Hough Transform Algorithm.

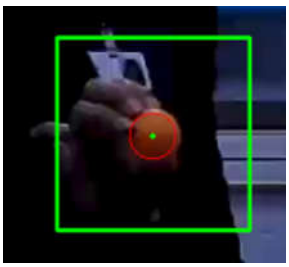


Figure 4.7: (a) The region of interest (ROI), the circumference and the centre of the detected ball

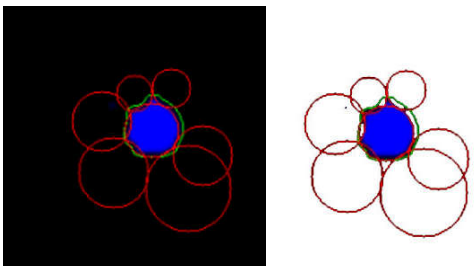
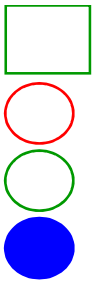


Figure 4.7: (b) The predicted ball, the circumference of OMLB and candidate circles

Where:



ROI

Candidate circles

Nearest Contour

OMLB

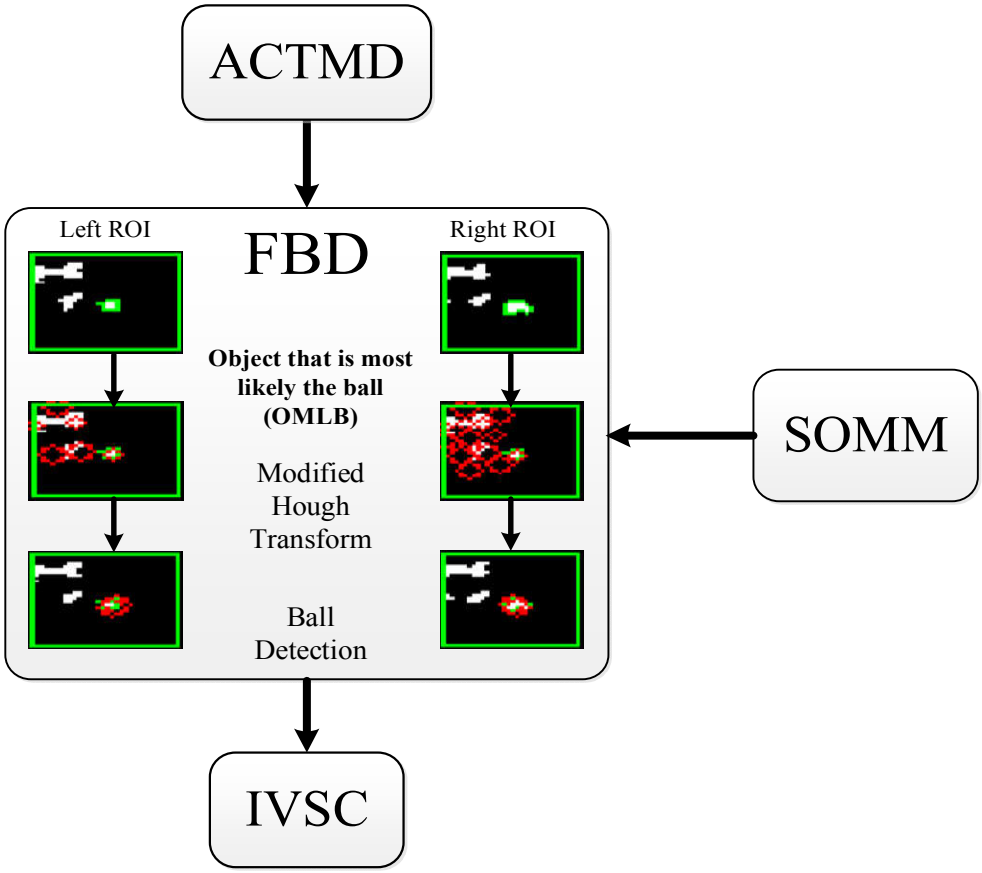
Figure 4.7: Identifying the OMLB

In FBD, the centre and radius of the ball is determined by finding the circle that is the best fit to the contour of the OMLB and nearest to the predicted location. To obtain a good fit, a large number of candidate circles need to be tried, but this has a cost implication. Therefore, the proposed strategy is to first generate many possible candidate balls but eliminate those that are of wrong sizes or orientations using *a priori* knowledge. To this end, the set of candidate circles produced by the Hough Transform are by applying the following four criteria:

1. **Location Assessment:** The centre of the candidate circle is enclosed by the contour of OMLB.
2. **Distance Assessment:** With the minimum Euclidean distance between the centre of that circle and the centre of predicted ball.

3. **Size Assessment:** With the smallest radius difference between the candidate circle and the predicted ball.
4. **Colour Assessment:** With the smallest average RMSE between the candidate circle and the previous detected ball in HSV colour.

The candidate ball that satisfies all above criteria is chosen to be a detected ball. Figure 4.8 illustrates the block diagram of the proposed ACTMD. To complement the block diagram, the process of FBD module is shown in Figure 4.9 below.



ACTMD: Adaptive Colour Thresholding and Motion Detection
SOMM: Second Order Motion Model
FBD: Feature based Ball Detection
IVSC: Inter-View Self-Correction

Figure 4.8: Block diagram of the proposed ACTMD

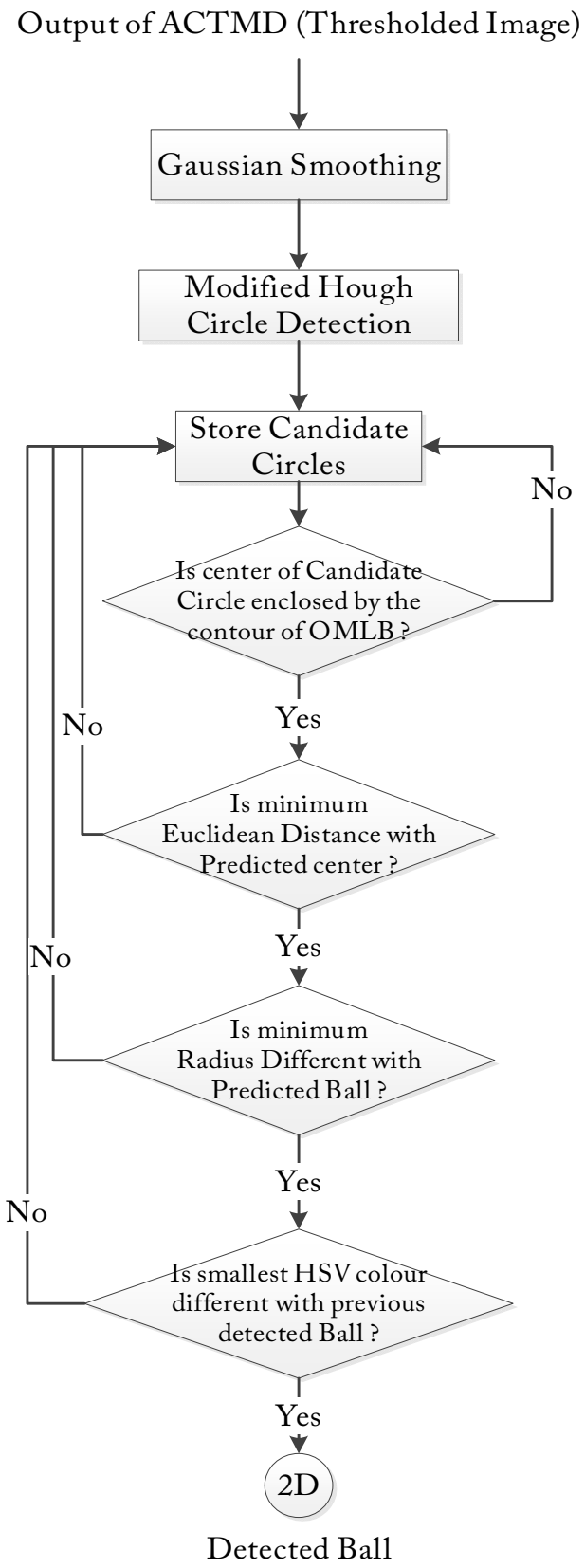


Figure 4.9 Flowchart of FBD module

To enable the system to check how good of the detection is, a detection score (S) is used for evaluation. It is calculated based on the Euclidean distance and RMSE between the detected ball and the predicted ball, and the predefined threshold values as:

$$\mathrm{S_n} = \begin{cases} if \quad (dist[DP] \quad < max_d) & \mathrm{S_n} = \mathrm{S_n} + 1; \\ if \quad (min_r \quad < r_D \quad < max_r) & \mathrm{S_n} = \mathrm{S_n} + 1; \\ if \quad (Avg\_RMSE\_HSV_D \; < \; acceptable\_HSV) & \mathrm{S_n} = \mathrm{S_n} + 1; \end{cases} \tag{4}$$

Where: S_n is the total number of detection score of the n^{th} frame. If the Euclidean distance between the centre of detected ball and the centre of predicted ball; $dist[DP]$ is less than the predefined distance; max_d, one point will be awarded. Similarly, if the radius of the detected ball; r_D is in between the predefined minimum radius; min_r and the maximum acceptable radius; max_r, one point will be awarded. Likewise, the average RMSE between the detected ball's HSV colour; $Avg\_RMSE\_HSV_D$ less than the acceptable threshold; $acceptable\_HSV$, one point will be awarded. As a result, if those deviations are smaller than or within the range of acceptable thresholds, one point each (three points maximum) will be awarded to its detection score for each satisfied criterion. This value will be used to guide the inter-view correction which will discussed in Section 4.2.4.

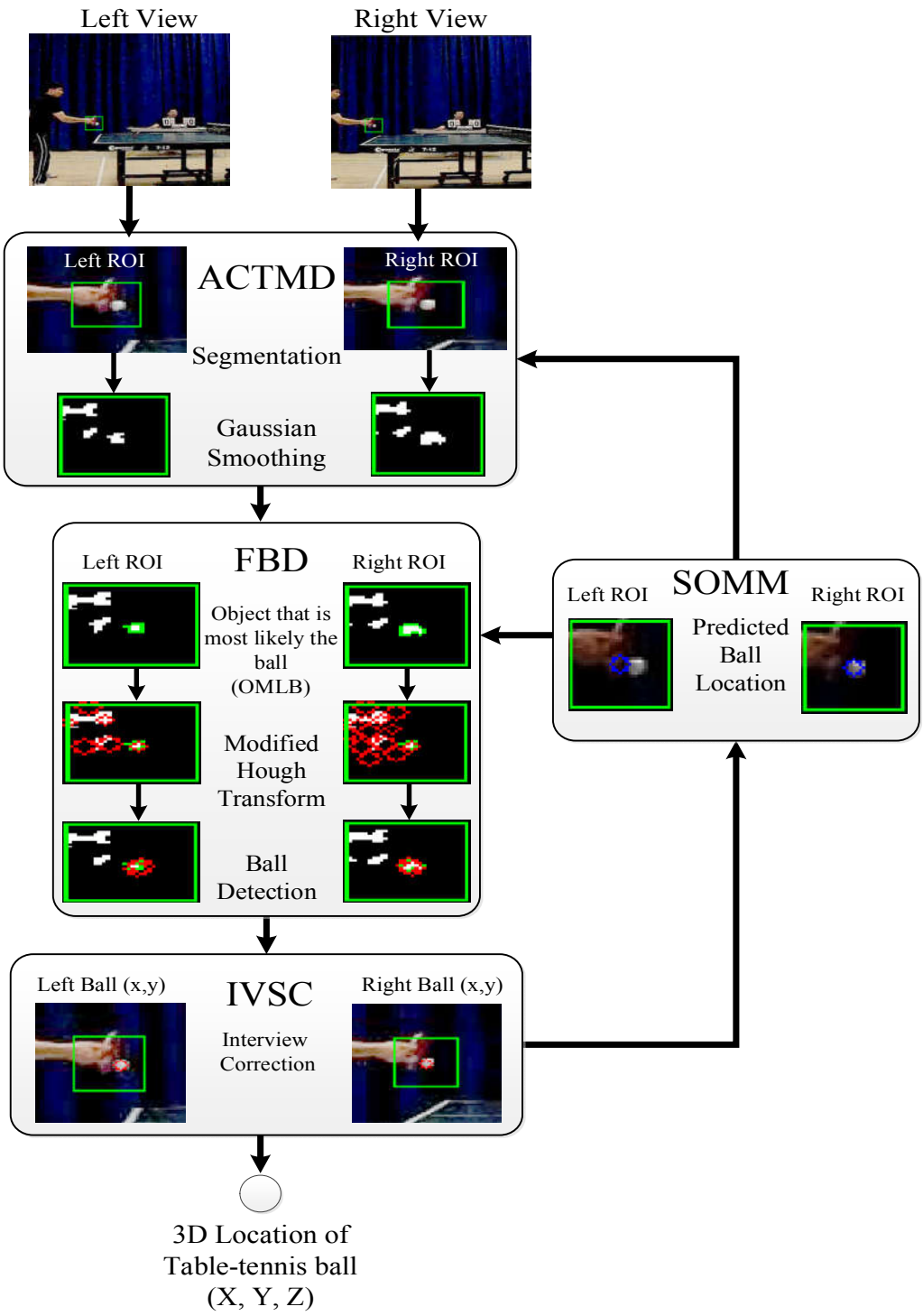
4.2.4 Inter-View Self-Correction (IVSC)

The IVSC module corrects the detected ball location if the detection score of one view is lower than the other one. As shown in figure 4.10, the test bed employs a pair of cameras and will detect balls from both left and right views. With this arrangement, when the ball in one view is undetected or wrongly

detected, its location can be estimated using the location of the detected ball from the other view and the known disparity value from a previous frame where balls from both views are successfully detected.

Likewise, if the ball is occluded by foreground object in one view, IVSC uses the positional information from another view to estimate the ball location of the current view. To illustrate the need and subsequent role that IVSC plays in the detection process, Figure 4.25 provided the results comparison and the usefulness of IVSC in sequence 3 as an example. The IVSC is not only applied in sequence 3. It applied throughout all the tested sequences throughout thesis.

In this way, the contribution of the IVSC module can significantly improve the detection accuracy and overcome the occlusion challenges. Finally, 3D ball's position is calculated using triangulation theory (Zhang, 2000). Detail explanation of stereo camera calibration and the calculation of 3D reprojection can be found in Chapter 3. By working together, the challenging task of tracking a high-motion ball during a match is successful, even though it was taken by two entry level high speed cameras. Figure 4.10 illustrates the block diagram of the proposed system which shows the interaction between all four main modules that explained above.



ACTMD: Adaptive Colour Thresholding and Motion Detection

SOMM: Second Order Motion Model

FBD: Feature based Ball Detection

IVSC: Inter-View Self-Correction

Figure 4.10: Block diagram of the proposed system

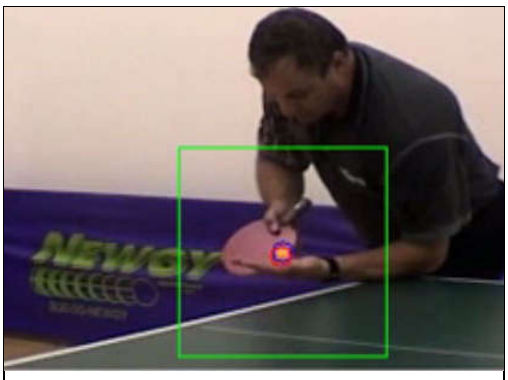
4.3 Tested Sequences and Experimental Results

The system was tested against three sequences from OUTTDB (Wong and Dooley, 2017). These sequences were particularly filmed to cover different scenarios such as a single view, (full and half table) Stereo-view videos. These were selected to test the developed algorithm on challenging conditions such as a sudden change of trajectory, occlusion, uneven illumination, scale variation, deformation, motion blur, noise and disappearance from the point of view.

4.3.1 Sequence 1: Results Discussion

The first sequence is a single-view video extracted from one of the demonstration video files on the web site of the Umpires & Referees Committee of the ITTF. This sequence is composed of 46 frames with (352*240) pixel resolution and a rate of 30 frames per second. Because of the low frame rate, object blurring and colour merging with the background can frequently occur. This sequence is selected to demonstrate the detection performance of the system during service because it was used in Wong and Dooley (2010) for testing their algorithm and it enables a performance comparison.

The figure 4.11 (a) and (b) shows an example of some experimental results of sequence 1. The blue and red circles show the predicted and detected ball positions of the ball. As the two circles are very close to each other, the blue circle is not clearly shown in the two figures. The green box represents the adaptive ROI which has an ability to automatically adjust its size according to the detection condition. **x** and **y** indicate the screen's coordinate (pixel) positions where the left top corner of the screen is (0,0) and **r** stands for radius of the ball.



Frame No. 18	x	y	r
Ground Truth (pixels)	**183**	**194**	**4**
System Result (pixels)	183	193	4
RMSE (2D)	**1.5 pixels**		
Detection Time	0.018 second		

Figure 4.11: (a): A service is about to start.

Frame No. 18	x	y	r
Ground Truth (pixels)	**204**	**161**	**5**
System Result (pixels)	205	163	4
RMSE (2D)	**2.7 pixels**		
Detection Time	0.018 second		

Figure 4.11: (b) The ball is about to be struck.

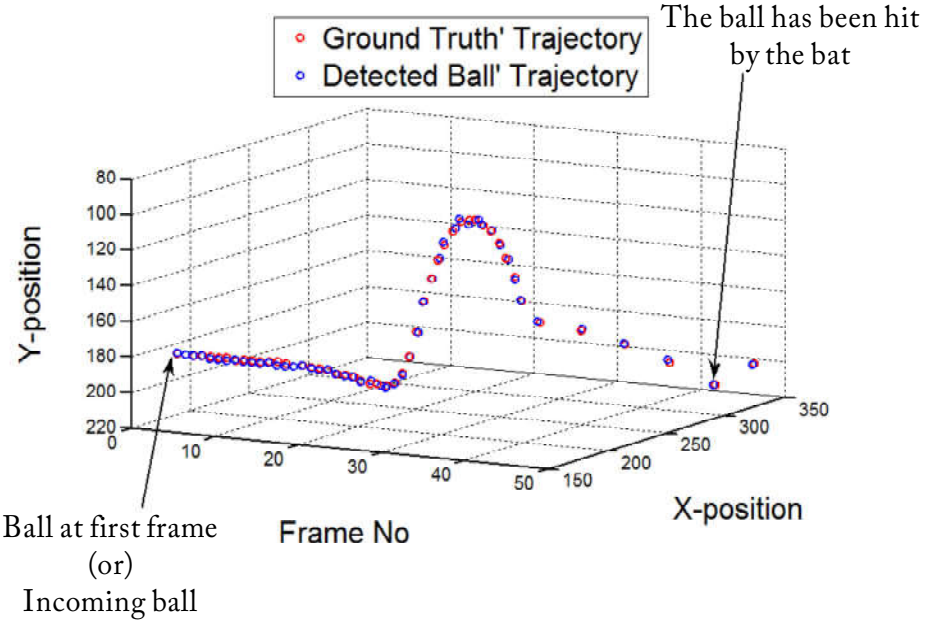
Figure 4.11: Example Frame in tested sequence 1

A summary of the sequence 1's detection results and a trajectory comparison of the ground truth and the detected ball where the ball is located at the player's palm and served is shown in Figure 4.12. In figure 4.12, the ground truth's trajectory is based on the set of ball locations identified by human volunteers on each frame of each view which is explained detail in Chapter 3. Since this research is targeted to be a real-time umpiring system, three key quantitative parameters, RMSE, Detection rate and Processing time have been used throughout this thesis as benchmarks to validate all the performance results. As can be seen in figure 4.12, it is clear from the comparison that the trajectories are highly aligned, indicating successful detection throughout the whole sequence. While the average radius of the ball in each frame is around 4.5 pixels, the system can detect the ball with RMSE 1.38 pixels. Since, the RMSE is less than the radius of the ball, the

detecting result is good enough for developing an umpiring system. The key features of the tested sequence 1 is shown in Table 4.1.

Table 4.1: Summary of the features of tested Sequence 1

Features of Tested sequence	Sequence 1
No of frames	46
Size of frame (pixels)	352×240
Capture rate	30 fps
Average radius of the ball	4.5 pixels
Ball colour	Orange
Key detection challenges	Low frame rate, object blurring, merging, occlusion



Quantitative Result	Total frames	Detection rate	Average radius of the ball	Average RMSE	Detection time
Sequence 1	46	100%	4.5 pixels	1.38 pixels	0.01 sec

Figure 4.12: Trajectory Comparison between Ground Truth and Detected Balls

To derive 3D location of an object, it requires the 2D result from at least two views. Therefore, a single view result alone is not enough to calculate the 3D location of the ball which is an essential information to know for umpiring a rally. Typical example is whether the ball is served above the level of the playing surface or behind the server's end line. For this reason, the second and third sequences are captured during real matches with a custom-made stereo camera to achieve 3D location of the ball.

4.3.2 Sequence 2: Results Discussion

The second sequence is composed of 400 frames with (512*384) resolution and a rate of 300 frames per second. Although it can achieve the whole table wider view, the size of the ball in this sequence appears much smaller and this increases the detection challenge. The following figure 4.13 shows one example frame of sequence 2.

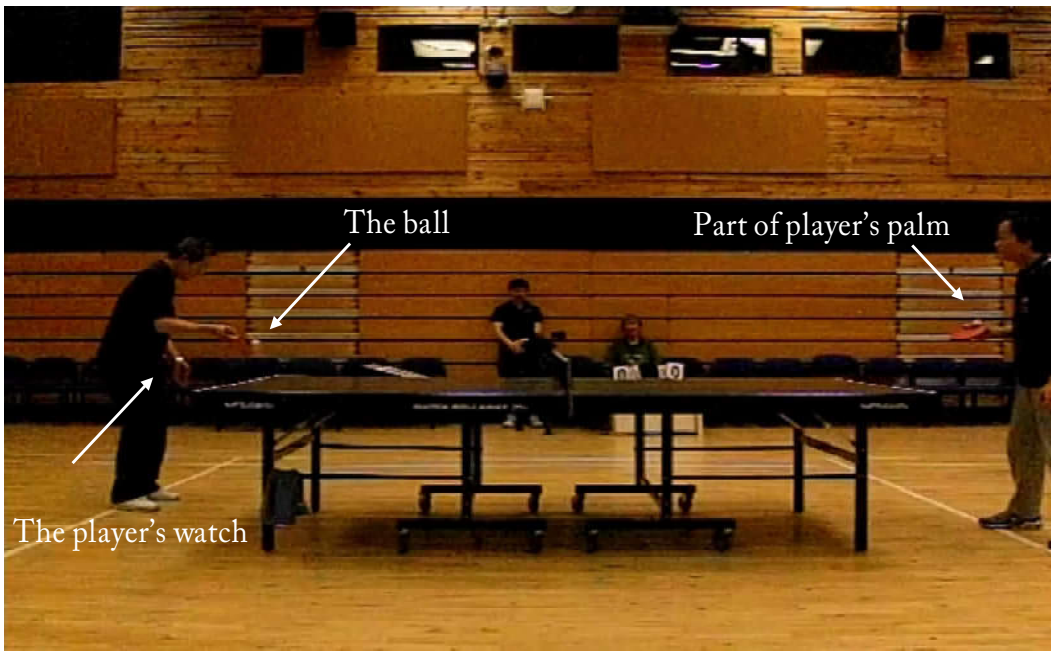
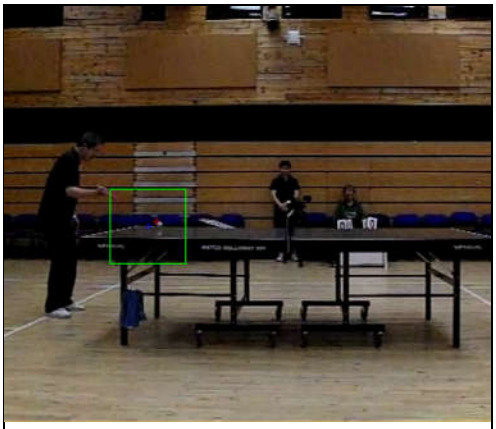


Figure 4.13: The full table view of tested Sequence 2

In sequence 2, the average radius of the image of ball appears as only 1.95 pixels. The complex background contains many white horizontal lines as well as ball-like objects, so this creates a challenging detection scenario including colour merging and shape confusion.

Frame No. 25	x	y	r
Ground Truth (pixels)	**157**	**219**	**2**
System Result (pixels)	157	219	2
RMSE (2D)	0.12 **pixels**		
Detection Time	0.059 second		

Figure 4.14: (a) Tested sequence 3: The ball is about to bounce on the table surface at Left Camera's View at Frame No: 25



Frame No. 25	y	r	r
Ground Truth (pixels)	**149**	**214**	**2**
System Result (pixels)	148	213	2
RMSE (2D)	1.5 **pixels**		
Detection Time	0.059 second		

Figure 4.14: (b) Tested sequence 3: The ball is about to bounce on the table surface at Right Camera's View at Frame No: 25

Figure 4.14: Example Frame in tested sequence 2

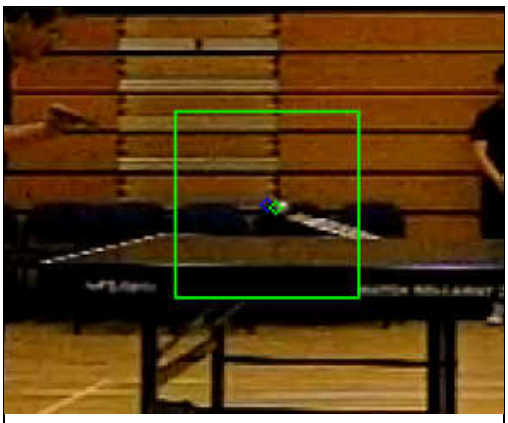
The table 4.2 to 4.5 show the left and right (x, y) screen coordinate (pixels) to (X, Y, Z) 3D reprojection results in centimetres (cm) where; X stands for the ball's horizontal running distance from the camera, Y indicates the ball's vertical height

from the camera and Z is the depth, which indicates how far between the ball and the left camera. R stands for the radius of the detected ball in (cm).

Table 4.2: Tested sequence 2: Ball 3D calculation results at Frame No 25

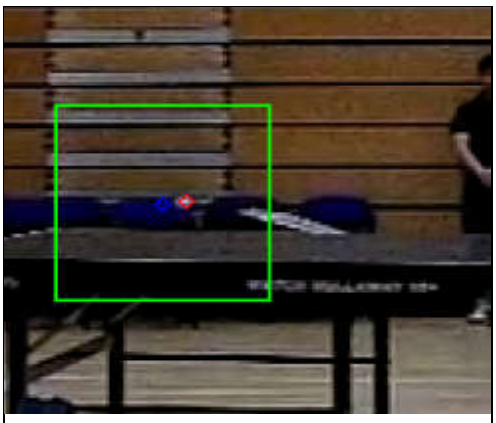
Real world 3D Measurement	X	Y	Z
Ground Truth	85 cm	56 cm	640 cm
System Result	82 cm	54 cm	619 cm
RMSE (3D)	21.3 cm		

In the figure 4.15 demonstrate the result of IVSC in where, the green circumference defines the recovery position of that mis-detected ball's location based on the positional information from the right view when the ball fails to be detected in in left view.



Frame No. 39	x	y	r
Ground Truth (pixels)	**175**	**211**	**2**
System Result	171	212	3
RMSE (2D)	3.7 **pixels**		
Detection Time	0.059 second		

Figure 4.15: (a) Green Corrected Ball in Left Camera's View Frame No: 39 (Corrected Result)



Frame No. 39	x	y	r
Ground Truth (pixels)	**164**	**207**	**2**
System Result	165	208	2
RMSE (2D)	2.2 **pixels**		
Detection Time	0.059 second		

Figure 4.15: (b) Red Detected Ball in Right Camera's View Frame No: 39 (Detected Result)

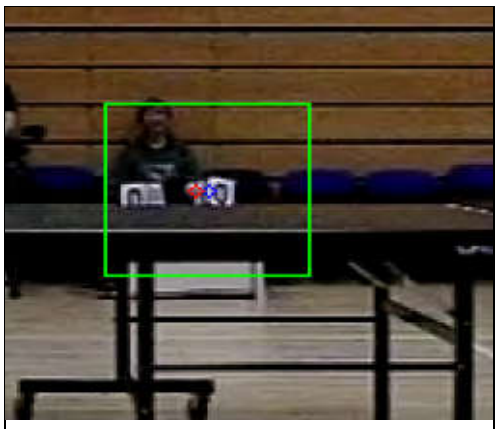
Figure 4.15: Example Frame in tested sequence 2

Table 4.3: Tested sequence 2: Ball 3D calculation results at Frame No: 39

Real world 3D Measurement	X	Y	Z
Ground Truth	63 cm	44 cm	580 cm
System Result	79 cm	53 cm	686 cm
RMSE (3D)	107 cm		

Frame No. 156	x	y	r
Ground Truth (pixels)	**339**	**217**	**3**
System Result (pixels)	339	217	2
RMSE (2D)	0.8 **pixels**		
Detection Time	0.059 second		

Figure 4.16: (a) Detected result of the ball when it is crossing scorecard at Left View Frame No: 156



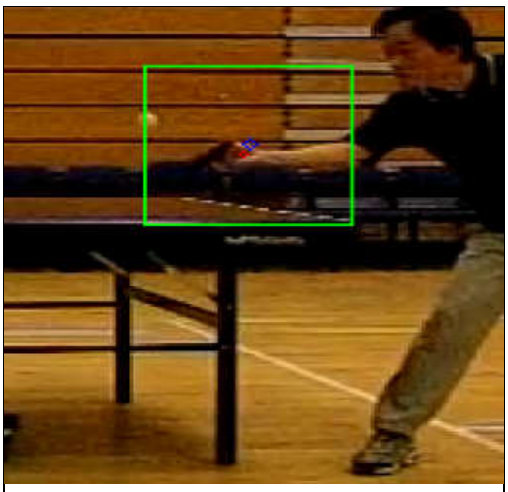
Frame No. 156	x	y	r
Ground Truth (pixels)	**322**	**212**	**2**
System Result (pixels)	321	212	2
RMSE (2D)	1.3 **pixels**		
Detection Time	0.059 second		

Figure 4.16: (b) Detected result of the ball when it is crossing scorecard at Right View Frame No: 156

Figure 4.16: Example Frame in tested sequence 2

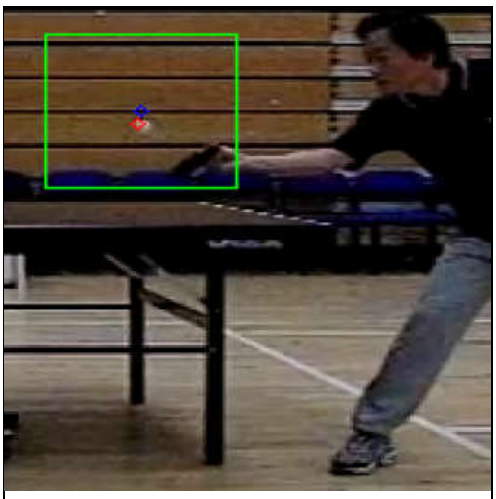
Table 4.4: Tested sequence 2: Ball 3D calculation results at Frame No: 156

Real world 3D Measurement	X	Y	Z
Ground Truth	55 cm	41 cm	490 cm
System Result	54 cm	40 cm	477 cm
Average RMSE	12 cm		



Frame No. 232	x	y	r
Ground Truth (pixels)	**395**	**192**	**3**
System Result (pixels)	417	201	3
RMSE	24.08 **pixels**		
Detection Time	0.059 second		

Figure 4.17: (a) Detected result of the ball after it got struck by the player in Left View Camera's Frame No: 232 (Incorrect detection)



Frame No. 232	x	y	r
Ground Truth (pixels)	**388**	**190**	**3**
System Result (pixels)	385	189	2
RMSE	3.6 **pixels**		
Detection Time	0.059 second		

Figure 4.17: (b) Detected result of the ball after it got struck by the player in Right View Camera's Frame No: 232 (Correct detection)

Figure 4.17: Example Frame in tested sequence 2

Table 4.5: Tested sequence 2: Ball 3D calculation results at Frame No: 232

Real world 3D Measurement	X	Y	Z
Ground Truth	125 cm	33 cm	662 cm
System Result	77 cm	22 cm	352 cm
Average RMSE	313 cm		

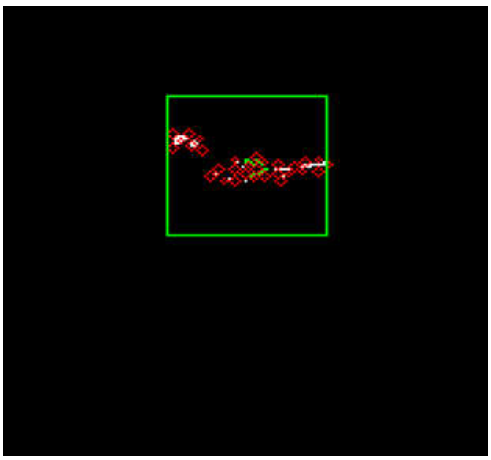


Figure 4.18: (a) Incorrect detection - Left Camera's View at Frame No: 232

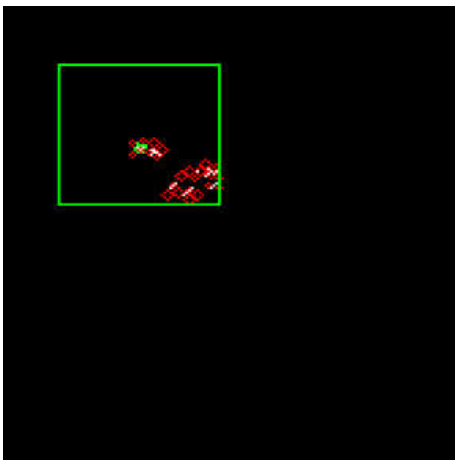


Figure 4.18: (b) Correct detection - Right Camera's at Frame No: 232

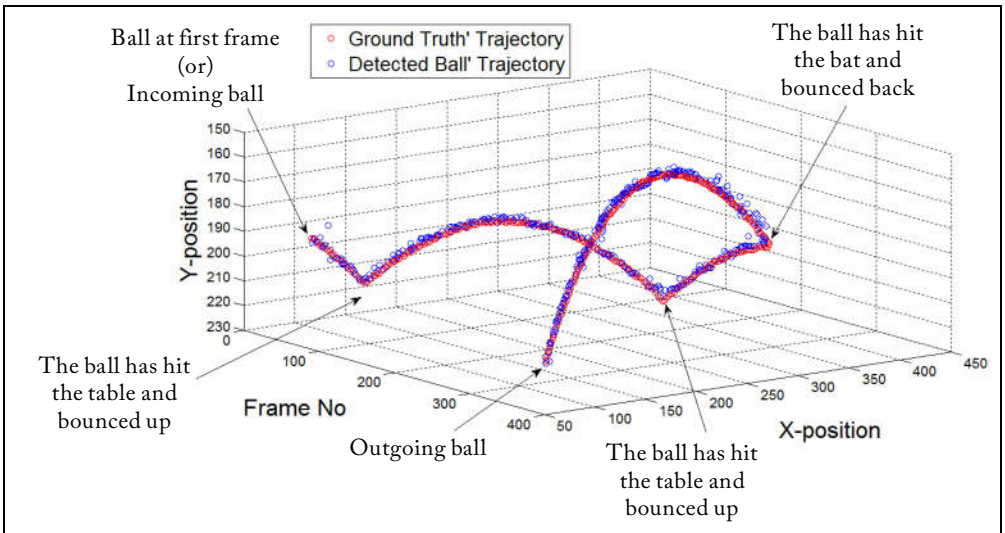
Figure 4.18: Example segmented result of the ball when it got struck by player

Figures 4.18 illustrates the Left and Right View's detected result of the ball after it got struck by the player. As mentioned above, incorrect detection can occur when the ball comes toward to the player side, or after it struct by the player. As can be seen in figures 4.18 not only the ball, but also some parts of player hand as well as the bat appear in the segmented results, as they all hold similar colour features and motion. However, IVCA module is really helpful in that situation and the system is able to recover and continuously track the ball with only occasionally a small deviation. A trajectory comparison of the ground truth (red) and the detected ball (blue) of sequence 2 where the ball has travelled from left to right is shown in Figure 4.19 below. In that figure, the ball has travelled from left to right. After that, it hits with the table and bounce up. When the ball comes towards the player, it gets struck and bounces back to the opposite direction until the end of the rally. It is clear from the comparison that the trajectories are highly aligned, indicating successful detection throughout the

whole sequence. A summary of the key features of the tested sequences 2 is shown in Table 4.6 below.

Table 4.6: Summary of the features of tested sequence 2

Features of Tested sequence	Sequence 2
No of frames	400
Size of frame (pixels)	512×384
Capture rate	300 fps
Average radius of the ball	1.95 pixels
Ball colour	White
Key detection challenges	Colour merging, Object reflection, Multiple moving objects, Occlusion, Complex background, Very small ball's size



Quantitative Result	Total frames	Detection rate	Average radius of the ball	Average RMSE	Detection time
Sequence 2	400	91%	1.95 pixels	1.9 pixels	59 ms

Figure 4.19: Trajectory Comparison between Ground Truth and Detected Balls

4.3.3 Sequence 3: Results Discussion

While the second sequence can show the full table view, the third sequence only shows the half length of the table. As it captures closer to the area of play (approximately 2 meters), the image of the ball appears bigger and clearer. The second sequence is composed of 200 frames with (512*384) resolution and a rate of 300 frames per second. Due to the low resolution of the camera, sequence 3 appears darker than the actual match scenery. However, this sequence is selected to showcase that the developed algorithm is robust enough to detect the ball among confusing objects which include ball like images, moving parts of referee's body and multiple light illuminations. Figure 4.20 (a) and (b) illustrate the before and after detection results in which the colour of the ball is merging with the background objects, yet the system can successfully detect the ball in the situation of colour merging, illumination and shape distortion.

Figure 4.20: (a) Tested sequence 3: Original Frame

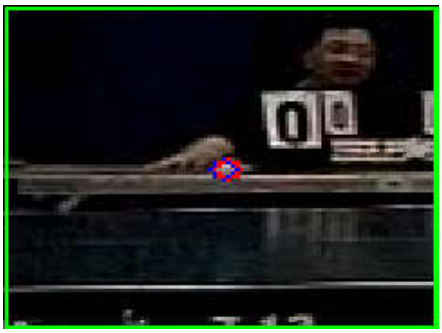


Figure 4.20 (b) Test sequence 3: Detection result

Figure 4.20: Comparison between before and after detection results

Figures 4.21 (a) and (b) demonstrate when the ball comes toward to the player and immediately after it got hit with the bat. In that moment, not only the ball but also the player's hand appears in ROI with similar motion. When the ball is about

to get hit, not only the ball but also the player's hand appears in the ROI and moves with similar motion. If the video is taken from a farther away location, the player's thumb which is holding a bat can appear as a similar shape and comparable size as ball and moving with similar motion in the ROI.

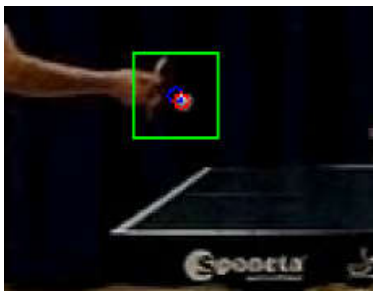


Figure 4.21 (a) The ball is about to get hit at Left Camera's View

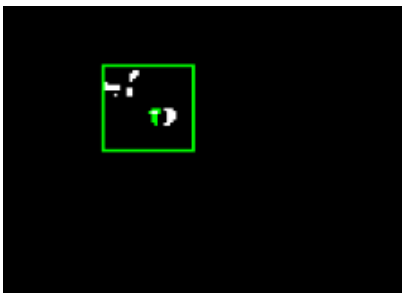


Figure 4.21 (b) Thresholded Right Camera's View Result

Figure 4.21: Tested sequence 3: Example Frame No: 149

If the system is solely depending on the MDA, this is the point where the system may fail to detect the ball. Similar situation is when the ball hits the net, the net will be vibrated, and this vibration will also be in the ROI. However, the developed system can detect this kind of difficult scenario with good detection results by the proposed adaptive segmentation approach. One of the advantages is in case detection fails by motion detection, the system can have another chance to segment again with CT by dynamically adapting and switching method. In here, it is required to discuss the choice of different orders motion model. According to the experimental results, if the algorithm is developed by a higher order motion model which relies on several previous detected ball locations;

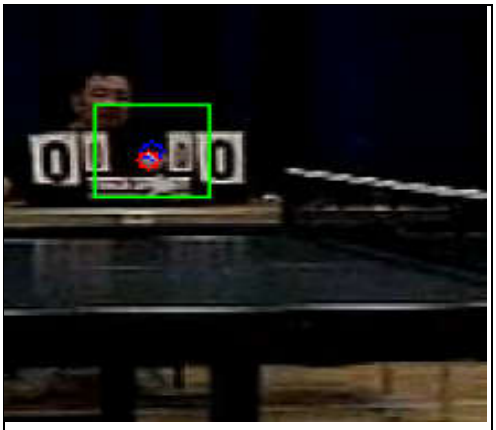
- the predicted direction tends to be the same as the previous direction and it is difficult to pick up the abrupt trajectory change;
- the system wrongly detects ball like objects on the player's body when the ball comes toward to the player and immediately after it got hit with the bat.

Therefore, the assumption is the higher order motion model has several delays in trajectory change whenever the ball bounces on the table or it is struck by the player. The analysis and simulation results indicate that the lower order motion model turns out to be the best fit for predicting the ball that has an abrupt trajectory change and, in SOMM, the predicted ball location is decided by a SOMM. Another challenging situation is when the ball is about to bounce, the reflection of white ball appears on the playing surface and it creates another ball like object in the ROI as can be seen in figure 4.21. Likewise, when the ball is travelling slowly or the point where it travels backward direction (bouncing back), the images of the ball in the background and current frame are partially overlapped and the thresholded result has two crescent shaped objects situated at equal distance from the predicted ball location as shown in figure 4.21 (b). Based on the travelling direction of ball, SOMM identify the right object and remove the crescent corresponding to the ball image in the background. The figure 4.22 to 4.23 show some experimental results of sequence 3's left view and right view and the table 4.7 shows the 3D calculation results of one example frame. As can be seen in figure 4.22, the white scorecard with number zeros create a challenging task in distinguishing the right ball among similar objects, yet the developed algorithm can provide the good detection results with those situations.

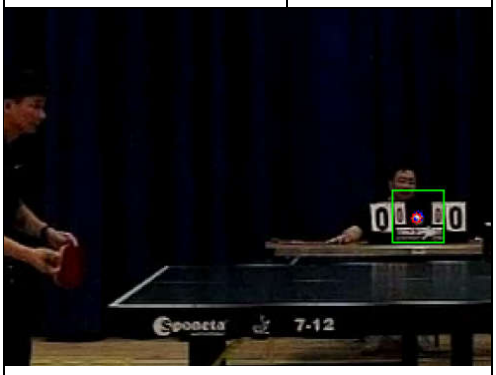
Frame No. 61	x	y	r
Ground Truth (pixels)	363	231	3
System Result (pixels)	363	231	3
RMSE (2D)	0.4 pixels		
Detection Time	0.048 second		

Frame No. 64	x	y	r
Ground Truth (pixels)	359	233	3
System Result (pixels)	358	232	3
RMSE (2D)	1.06 pixels		
Detection Time	0.048 second		

Figure 4.22 (a) Tested sequence 3: The ball is crossing the scorecard at Left Camera View



Frame No. 61	x	y	r
Ground Truth (pixels)	320	231	4
System Result (pixels)	319	231	3
RMSE (2D)	0.76 pixels		
Detection Time	0.048 second		



Frame No. 64	x	y	r
Ground Truth (pixels)	314	232	3
System Result (pixels)	313	232	3
RMSE (2D)	0.76 pixels		
Detection Time	0.048 second		

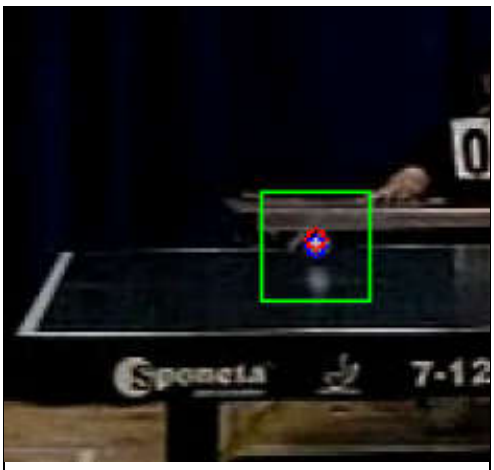
Figure 4.22 (b) Tested sequence 3: The ball is crossing the scorecard at Right Camera View

Figure 4.22: Example Frame of Tested sequence 3

Table 4.7: Tested sequence 3: Ball 3D calculation results at Frame No: 64

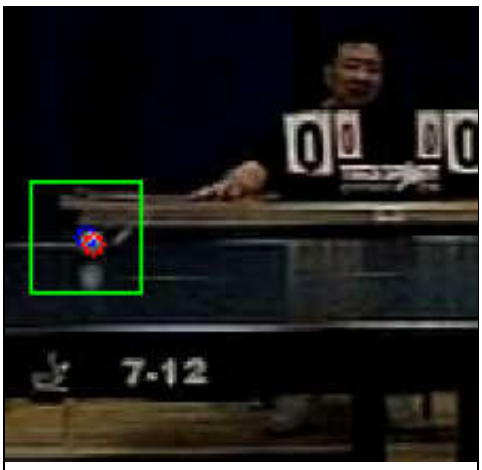
Real world 3D Measurement	X	Y	Z
Ground Truth	40 cm	30 cm	283 cm
System Result	39 cm	30 cm	283 cm
RMSE	0.5 cm		

In these figures, the green box represents the cropped ROI, the blue and red circle indicate the predicted and detected positions of the ball. The small letter **(x, y)** indicate the screen's coordinate (pixel) positions where the left top corner of the screen is (0,0) and **r** stands for radius of the ball. The table 4.8 and 4.9 show the left and right (x, y) screen coordinate (pixels) to (X, Y, Z) 3D reprojection results in centimetres (cm) where; X stands for the ball's horizontal running distance from the camera, Y indicates the ball's vertical height from the camera and Z is the depth, which indicates how far between the ball and the left camera. R stands for the radius of the detected ball in (cm). Detail explanation of 2D to 3D projection can be found in Chapter 3, Section 3.5.3.



Frame No. 101	x	y	r
Ground Truth (pixels)	277	259	3
System Result (pixels)	276	258	3
RMSE (2D)	0.6 pixel		
Detection Time	0.043 second		

Figure 4.23 (a) The ball is about to bounce on the table surface at left Camera View



Frame No. 101	x	y	r
Ground Truth (pixels)	233	257	3
System Result (pixels)	232	257	3
RMSE (2D)	0.96 pixel		
Detection Time	0.043 second		

Figure 4.23 (b) Tested sequence 3: The ball is about to bounce on the table surface at right Camera View

Figure 4.23: Tested sequence 3 at Frame No: 101

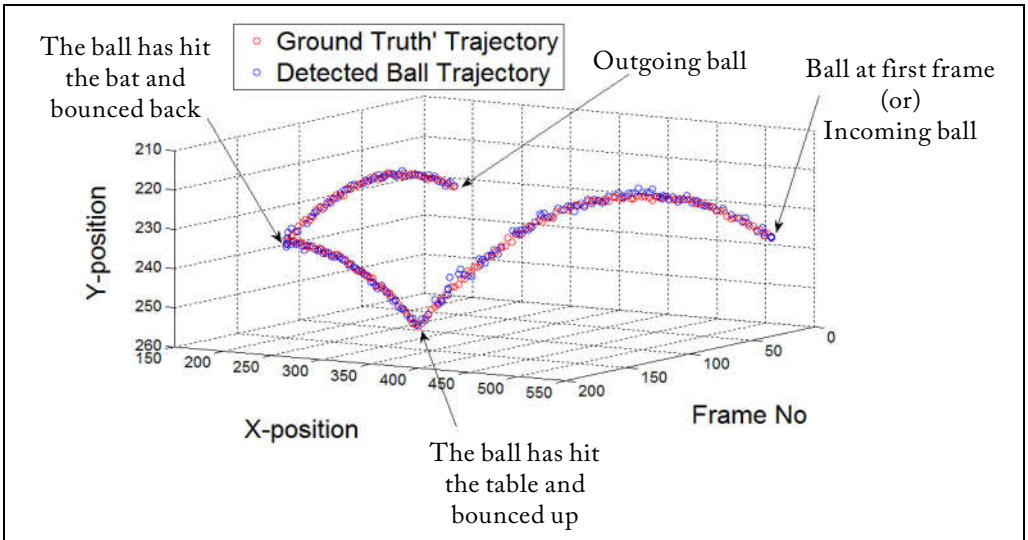
Table 4.8: Tested sequence 3: Ball 3D calculation results at Frame No: 101

Real world 3D Measurement	X	Y	Z
Ground Truth	8 cm	41 cm	288 cm
System Result	8 cm	40 cm	288 cm
RMSE (3D)	0.6 cm		

A summary of the key features of the tested sequence 3 is shown in Table 4.9 below. Figure 4.24 shows the trajectory comparison between the ground truth and the detected balls where the ball has travelled from right to left.

Table 4.9: Summary of the features of tested sequence 3

Features of Tested sequence	Sequence 3
No of frames	200
Size of frame (pixels)	512×384
Capture rate	300 fps
Average radius of the ball	3.4 pixels
Ball colour	White
Key detection challenges	Colour merging, illumination, shape distortion, object reflection, multiple moving objects, occlusion

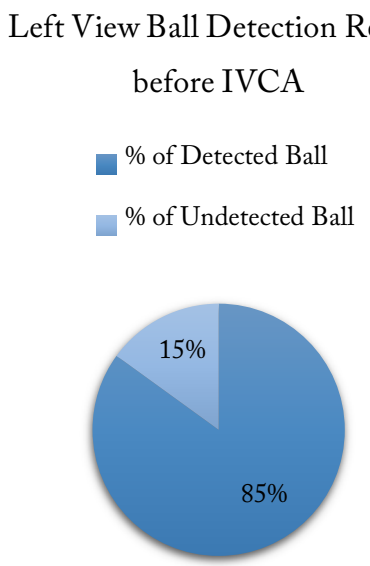


Quantitative Result	Total frames	Detection rate	Average radius of the ball	Average RMSE	Detection time
Sequence 3	200	95.5%	3.4 pixels	1.6 pixels	0.047 sec

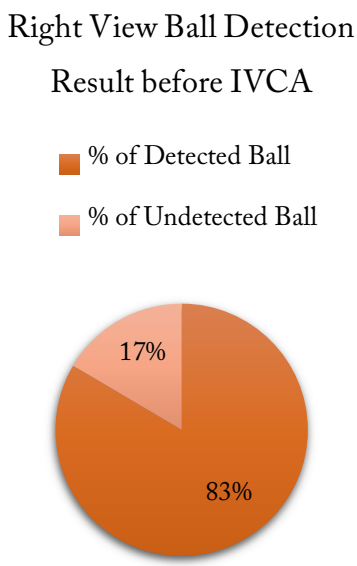
Figure 4.24 Trajectory Comparison between Ground Truth and Detected Balls

In figure 4.24, the ball is coming from the right corner of the frame at the starting point of the sequence 3. After that, it hits the table and bounce up. When the ball comes toward to the player, it gets struck and bounces back in the opposite direction.

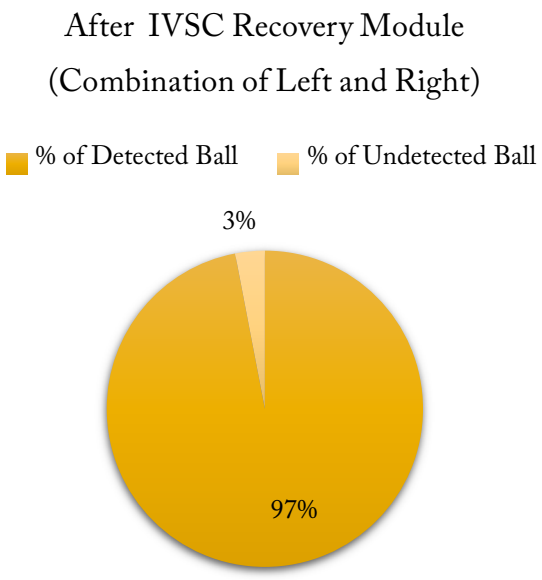
According to the quantitative result, while the average radius of the ball is around 3.4 pixels, the average RMSE of 2D calculation results is around 1 to 2 pixels. This contribute the actual 3D error of RMSE distance is less than 1 cm. If the RMS error between the detected location and the ground truth is less than the diameter of the ball, the detection is assumed to be correct and this provide enough information to umpire a rally. Moreover, the benefit of employing two cameras (in sequence 2 and 3) rather than a single camera (in sequence 1) is when the ball in one view is occluded or undetected, its location can be estimated by using the location of the detected ball from the other view and the known disparity value from a previous frame where balls from both views are successfully detected. With this arrangement, it can significantly improve the detection performance. The figure 4.25 illustrates the result comparison of before and after applying IVSC. In figure 4.25 (a) and (b), the detection rate of left single view is 85% as well as the detection rate of right single view is 83%. After applying IVSC module, each view's undetected ball location was estimated by using the detected ball location from the other view and the final detection results can be better as shown in figure 4.25 (c).



(a) Left View Detection Result before applying IVC module



(b) Right View Detection Result before applying IVC module



(c) Left and Right Detection Rate after IVSC

Figure 4.25: Effectiveness of IVSC in Sequence 3

4.4 Performance Comparison

Table 4.10 (a) and (b) show the performance comparison between three sequences including the results comparison with (Wong and Dooley, 2010). The proposed system outperformed it on both detection rate and speed. Among the three tested sequences which have different filming scenarios, the detection results in sequence 1 is the best among the other sequences. The reason is the ball size in sequence 1 appears bigger than other sequences because it was taken closer and focused only the service part. The orange ball in sequence 1 is also an outstanding colour feature compared with the white one in the other two sequences. Although the sequence 2 provided the full table stereo-view which were filmed from a long distance, they do not have enough depth resolution to calculate precise real-world coordinates (X, Y, Z) of objects and it was affected in 3D reprojection. The worse is the resolution of the camera is low and depth information becomes unstable. This contributes to 3D reprojection; Z (Depth) error can be more than 100 cm. Moreover, the original size of table tennis ball is small which is only 4 cm in diameter and when capturing the full table view, the ball appears very small in video as 1.9 pixels, 0.005% of the whole frame. As the bigger ball and clearer view of the play can provide higher detection rate, it is better to place the cameras at closer position. The second sequence is taken around 4 meters further away from the camera and the table tennis table and the third sequence is taken around 2 meters. As a result, the average radius of the ball become bigger than the sequence 2 with the RMSE of 2D calculation results is reduce from 4 pixels in some frames to 1 pixels. Nevertheless, the average RMSE

between the detected location and the ground truth is less than the diameter of the ball. While the processing time is good enough for real-time umpiring, the proposed system can accurately track the balls with a high success rate, despite occlusion, colour merging, reflection, blurring and shape distortion occurring in all these tested sequences. Even with the very challenging Sequence 2, which has a complex background and a very small ball, the detection rate is still over 91%. The only occasions where the detection failed were when the ball was occluded or severely merged with the background.

Table 4.10 (a): Performance comparison between three tested sequences

Quantitative Result	Sequence 1	Sequence 2	Sequence 3
No of frames	46	400	200
Size of frame (pixels)	352×240	512×384	512×384
Detection rate	100%	91%	95.5%
Average radius of the ball	4.5 pixels	1.95 pixels	3.4 pixels
Average RMS error (X, Y, R)	1.38 pixels	1.9 pixels	1.6 pixels
Average Processing time	0.018 sec	0.059 sec	0.047 sec

Table 4.11 (b): Performance comparison between sequence 1 and (Wong and Dooley, 2010)

Quantitative Result	Sequence 1	(Wong and Dooley, 2010)
No of frames	46	46
Size of frame (pixels)	352×240	352×240
Detection rate	100%	98%
Average radius of the ball	4.5 pixels	4.5 pixels
Average RMS error (X, Y, R)	1.38 pixels	-
Average Processing time	0.018 sec	0.1 sec

4.5 Summary

This chapter began with describing the strength and weakness of the existing ball detecting and tracking methods and identify the requirement for developing a new umpiring system. It was followed by the proposed algorithm which is designed with effective strategies. According to the experimental results, the assumption is made to film at enough depth to reproject the 3D results. In these experiments, a stereo vision is formed by pairing up two cameras and if the ball is occluded in one view (either left or right), the algorithm has a mechanism to recover by positional information from another view. However, there is a possibility that the ball can be totally blocked by the player, the bat, clothing, or for other reasons in both cameras' view. Especially at the point where the ball flies' closer to the player before it gets struck. Although the predicted ball position can be used to recover for a few frames, the prediction will be failed if detection is missed in several consecutive frames. One approach to overcome occlusion is to scale up the system by adding more cameras to monitor the ball at different angles. However, this requires a high degree of co-ordination between different views and an ability to resolve conflicts when inconsistent information is acquired. Moreover, adding more cameras will increase the processing workload and time. To tackle this problem, a multi-agent-based ball tracking system will be proposed in coming Chapter 5 by developing the system as artificial intelligence system. By this way, each camera pair can be associated with an agent that can independently detect and track the ball by parallel in different views and can improve the detection performance.

Chapter 5

A Scalable Multi-View Tracking System (Contribution Chapter)

5.1 Introduction

To improve the detection performance of the previously developed ball detection algorithm in Chapter 4, this chapter presents the design and development of block 5 (Multi-view Tracking), part of the proposed framework which has presented in Chapter 1, figure 1.2. While adding more cameras will increase the chance of detection, it is financially and technically challenging due to the limited number of cameras to form stereo visions, and the processing workload and time. As mentioned in Chapter 1, the system is intended to be developed with minimum hardware requirement, user friendly camera setup and ease of installation to fulfil the aim of a widely used low-cost solution. Therefore, for capturing different type of sequences, various camera distances and different camera setups were experimented with such as:

- Single-view Video
- SBS Stereo-view (Full-Table)
- SBS Stereo-view (Half-Table)
- FTF Multi-view (Half-Table)

as described in Chapter 3. The single view, SBS Stereo-view (full and half table) were tested in the Chapter 4 and the results comparison can be found in Section 4.4, showing that each setup can have an impact on the ball detection accuracy and the cost of the system. In fact, the nearer the camera's distance with the play, the clearer is the achievable resolution, and the camera's setup height should be enough to capture the whole table surface. Therefore, the tested sequences in this chapter were filmed by placing cameras at a closer location with the playing surface to achieve a sufficient depth as shown in figure 5.1.

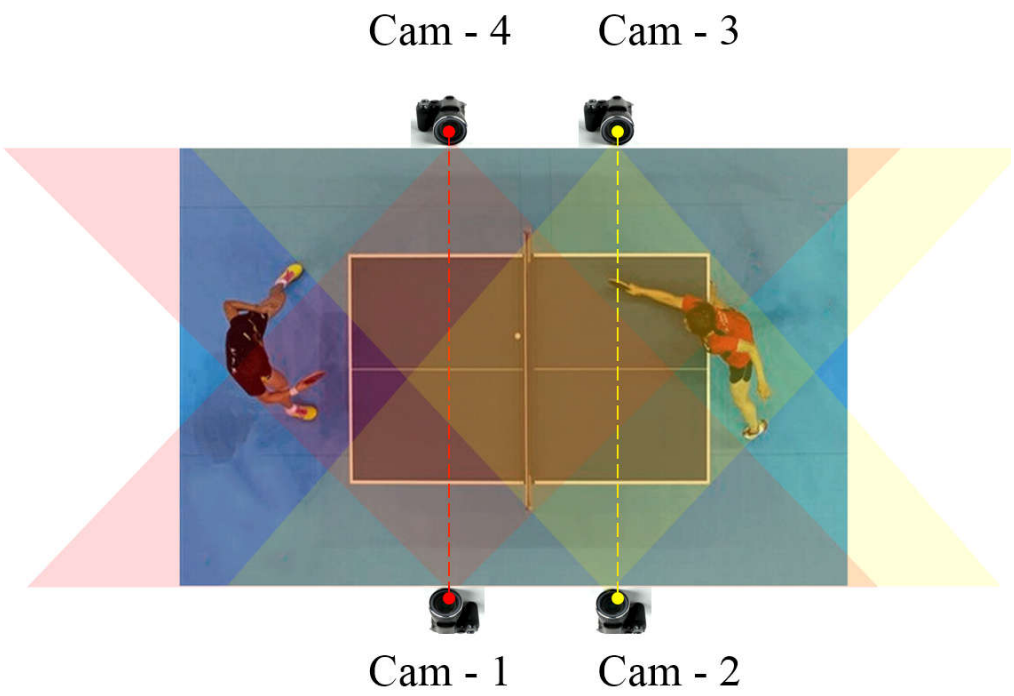


Figure 5.1: FTF Stereo Camera Set up

The detailed calibration process can be found in Chapter 3. In the traditional way of arranging stereo cameras (SBS setup), it requires two cameras to form one stereo-view and four cameras can only give two stereo-views. In this camera setup, the 3D location of ball can be derived by pairing two cameras next to each

other (SBS) or facing opposite (FTF). In this way four stereo-views can be obtained by using only 4 cameras, which is more cost effective. Moreover, this camera setup is also helpful in difficult ball detection situations where the captured image of the ball can be confused with nearby or background objects during a match. Example images of these challenging situations are shown in Figure 5.2.

Ball blurry | Ball reflection | Complex background

Colour merging | Blurry ball with moving hand and bat | Multiple moving objects (ball and spectators)

Figure 5.2: Ball Detection Challenges

If multiple opposite facing cameras monitor the playing surface, they can obtain different views of ball from different angles and provide higher detection opportunities in estimating the location of the ball by utilising the results from the opposing views. For example, if the image of the ball is merged with the

complex background in one view, it can be recovered by using the information from the opposite view. As a result, the chance of object confusion is reduced, and this can improve the detection rate which is essential in umpiring. While the multi-view approach strengthens the detection performance, it also has its own weakness in terms of data processing and memory requirements. Since these cameras are set up to monitor a closer view of the play, the output video cannot capture the whole scene but only a portion of the scene. The ball's trajectory also breaks up and requires a mechanism to join a trajectory from separate views. It increases computation due to synchronising and processing data from multiple cameras.

To tackle these problems, this chapter presents a novel framework of a multi-view ball tracking system. It addresses the multi-view detection problems by integrating the previously developed ball detection algorithm with a Multi-Agent System (MAS). The proposed design of the MAS enables the system to be computationally effective, lower in cost, portable and suitable for umpiring purposes. The remainder of the chapter is organised as follows: Section 2 briefly discusses the configuration of the cameras and the MAS architecture. It is followed by a detailed explanation of the implemented framework in Section 3. The techniques for compensating measuring errors are presented in Section 4. Experimental results and performance comparison are given to verify the effectiveness of the proposed algorithms in Section 5, while Section 6 discusses the limitation of the system and concludes the whole chapter.

5.2 Compensating Measuring Errors

While the proposed camera configuration can provide more robust ball detection, experiments have shown that aligning these cameras is very difficult and time consuming. Any misalignments could lead to errors in the derived 3D position of the detected ball, which will affect the accuracy of the overall umpiring system. For this reason, an Error Model was developed to compensate for measurement errors, which can be represented by 3D vectors. Various experimental results and analyses showed that the measurement error exhibited a non-linear relationship to the ball location as shown in figure 5.3 and 5.5. Therefore, quadratic surfaces were chosen to model the measuring errors. The measurement error data were fitted to the surfaces using a standard Multivariate Polynomial Regression. The equation of the error model is represented by equation (1):

$$\boldsymbol{E}_{(x,y,z)} = \boldsymbol{F}_{(x,y,z)}\ \hat{\imath}, \boldsymbol{G}_{(x,y,z)}\,\hat{\jmath}, \boldsymbol{H}_{(x,y,z)}\,\hat{k} \qquad (1)$$

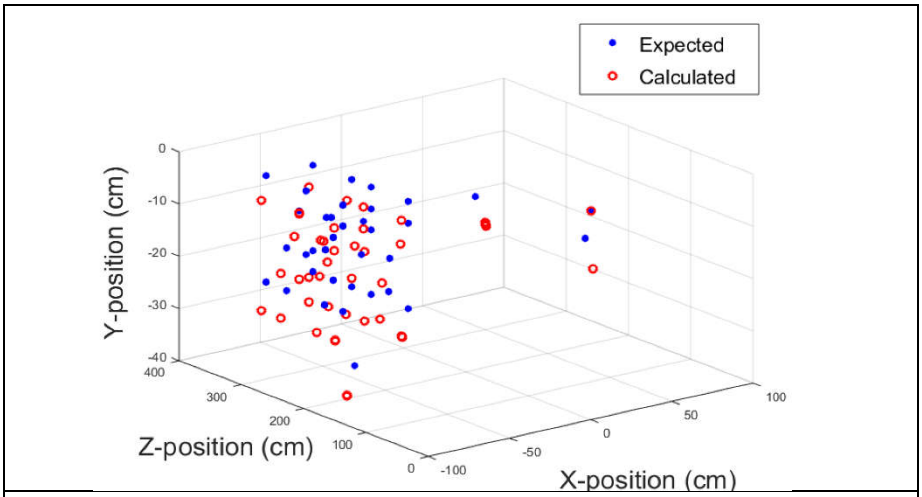
where $\boldsymbol{E}_{(x,y,z)}$ is the 3D error vector and $\boldsymbol{F}_{(x,y,z)}$, $\boldsymbol{G}_{(x,y,z)}$, $\boldsymbol{H}_{(x,y,z)}$ are functions determining the magnitudes of the $(\hat{\imath}, \hat{\jmath}, \hat{k})$ components respectively, and (x, y, z) are the measured ball location. In other words, the error model takes the 3D ball location as input and produces the error vector for that ball location. Equations (2), (3) and (4) define the quadratic surfaces of $\boldsymbol{F}_{(x,y,z)}$, $\boldsymbol{G}_{(x,y,z)}$, $\boldsymbol{H}_{(x,y,z)}$ respectively, where a_n, b_n, c_n, d_n, e_n, f_n, g_n, h_n, i_n and j_n are coefficients of the Quadratic surfaces, for n = 1, 2 and 3.

$$\boldsymbol{F}_{(x,y,z)} = a_1x^2+b_1y^2+c_1z^2+d_1xy+e_1xz+f_1yz+g_1x+h_1y+i_1z+j_1 \qquad (2)$$

$$\boldsymbol{G}_{(x,y,z)} = a_2x^2+b_2y^2+c_2z^2+d_2xy+e_2xz+f_2yz+g_2x+h_2y+i_2z+j_2 \quad (3)$$

$$\boldsymbol{H}_{(x,y,z)} = a_3x^2+b_3y^2+c_3z^2+d_3xy+e_3xz+f_3yz+g_3x+h_3y+i_3z+j_3 \quad (4)$$

To prevent overfitting, a small subset of 30 points was randomly selected from some known positions on the checkerboard as training data and 45 "unseen" points were chosen for testing. The detailed explanation of calibration with a double-sided checkerboard and deriving a large set of reference points on the checkerboard can be found in Chapter 3. Figures 5.3 and 5.5 show the 45 uncompensated calculated (red) and expected (blue) ball locations for the pairs of cameras (1 and 4) and (2 and 3). Figure 5.4 and 5.6 show the results of corrected ball locations after error compensation is applied. The average Euclidean distance between the calculated and expected positions reduced from 6.9 cm to 0.8 cm for one stereo pair (camera 1 and 4) and 10 cm to 0.9 cm the other stereo pair (camera 2 and 3). The experimental results from these tests indicate the developed error model can be used to significantly reduce the measuring errors. Although small magnitude errors still exist, the multi-view ball tracking system, which is to be explained in next section, can tolerate these errors.

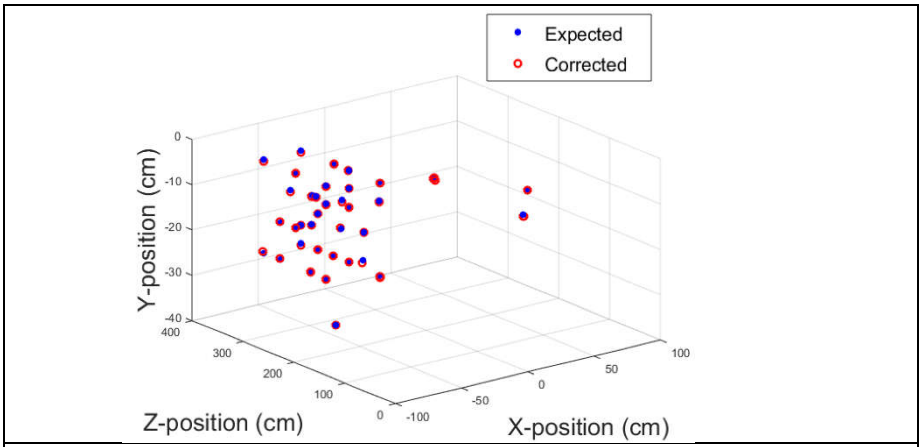


Euclidean Distance Error (Expected vs Calculated)

3D positions of the ball

Minimum Error	Maximum Error	Average Error	Standard Deviation
4.8 cm	10.2 cm	6.9 cm	0.9 cm

Figure 5.3: Uncompensated ball positions

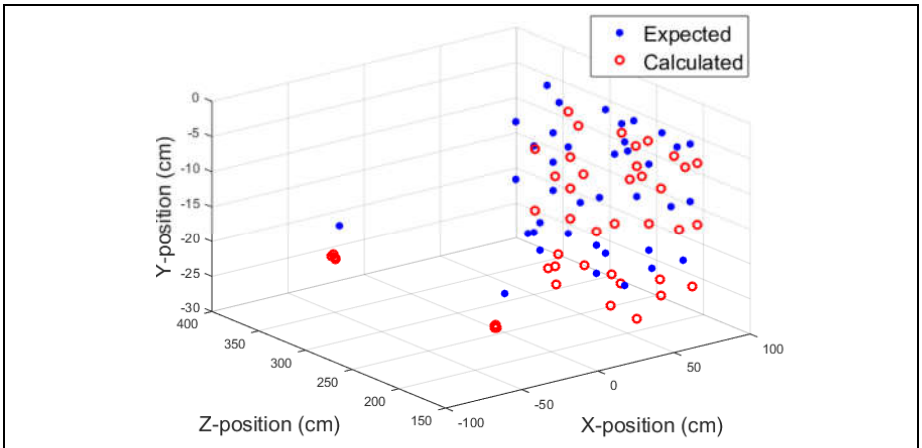


Euclidean Distance Error (Expected vs Calculated)

3D positions of the ball

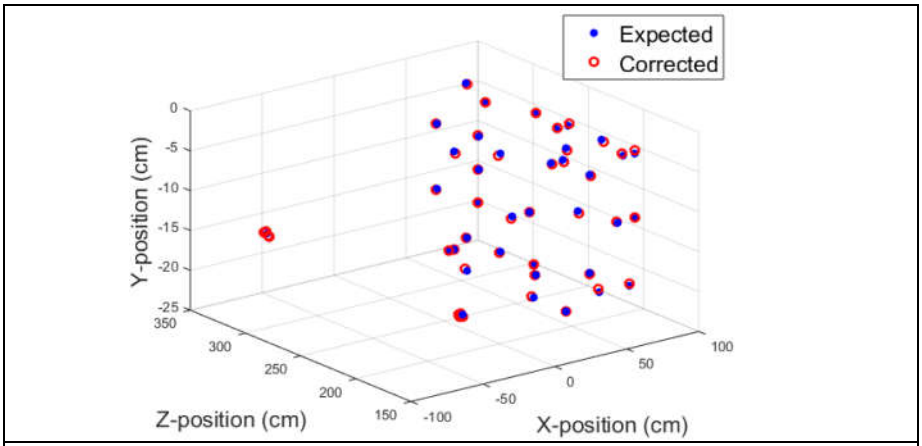
Minimum Error	Maximum Error	Average Error	Standard Deviation
0.2 cm	3.1 cm	0.8 cm	0.4 cm

Figure 5.4: Compensated ball positions



Euclidean Distance Error (Expected vs Calculated) 3D positions of the ball			
Minimum Error	Maximum Error	Average Error	Standard Deviation
7.1 cm	13.5 cm	10 cm	1.7 cm

Figure 5.5: Uncompensated ball positions



Euclidean Distance Error (Expected vs Calculated) 3D positions of the ball			
Minimum Error	Maximum Error	Average Error	Standard Deviation
0.1 cm	3.9 cm	0.9 cm	0.6 cm

Figure 5.6: Compensated ball positions

5.3 Multi-View Ball Tracking System

Although capturing images of an object from multiple angles can increase the chance of detection, careful consideration is needed in designing the configuration of the multi-view system, e.g., how many cameras are employed, where each camera is placed and how they are paired with other cameras to derive the 3D position of the object. To balance between cost and accuracy of detection, it was decided to employ four cameras, and they are placed at the positions as shown in figure 5.6. Four stereo-views was achieved by using four cameras. The 3D positions of the objects can also be derived from either the opposite facing pairs, i.e., FTF or the SBS pairs. The detail methodology of 3D position derivation and verification process can be found in Section 3.5.

In the figure 5.8, the four cameras jointly track the ball in the space within which the table and the players are situated. Each camera covers approximately two thirds of the length of the table but with a distinct perspective. With this arrangement, the main playing region is jointly covered by the four cameras and the coverage is overlapped around the net region, where attention is needed for umpiring. As each individual camera does not have to cover the entire table, the cameras can be placed nearer to the objects of interest (e.g. ball and table) so that better depth resolution can be achieved when deriving their 3D positions, and the objects also appear bigger in the views. The benefit is that the cameras can obtain distinct perspectives of the scene and can improve the detection.

Because of the workload in handling a huge amount of incoming data, it is necessary to develop a ball tracking framework which can facilitate tasks on a

multi-view system such as figuring out which camera pair will be used in analysing, when and where to turn on and turn off these cameras, and the results from which camera pairs will be set as higher priority among the detection results. As there are many observations such as sub-tasks for the system must be made, the choice of developing a system as a MAS turned out to be the best solution. A MAS can effectively manage the data generated by the multiple cameras situated at distinct positions. Since they all are linked by the MAS, the individual detection results can be jointly used to construct the whole trajectory. Moreover, a large amount of data from multi-view cameras can be processed in parallel and produce the result within a short period of time. For example, two or more agents could be used as Ball Detection Agents (BDA1, BDA2, etc...). These BDAs can detect the ball according to their point of view and can interact with each other in constructing a complete trajectory. By working together, the agent system can simultaneously make several observations. In the case of one ball detection agent not functioning, it will not affect the entire system and the other agents can continue to do their perspective task. Moreover, the developed system can be scaled up with more cameras to monitor the ball at different angles, in either a narrower or wider view. In this way, the system can overcome various detecting and tracking challenges in real match scenes and provide computational efficiency. In the implemented system, each camera pair is associated with a computing device (Agent) that can independently detect and track the ball in its associated views. The system is composed of two main agents:

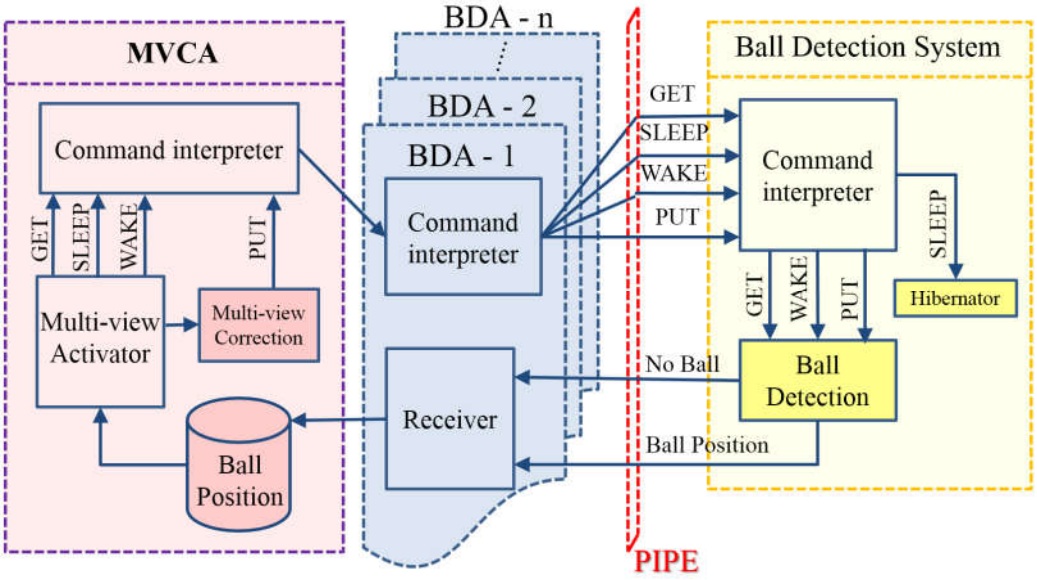
- Ball Detection Agent (BDA)
- Multi-View Correction Agent (MVCA)

The BDAs detect the ball based on the same principle as the ball detection system which was presented in the Chapter 4 and showed robust ball detection, but only considered the conventional SBS camera configuration. In this chapter, the system was modified to derive the ball's 3D position from opposite-facing cameras. Furthermore, as the ball detection system was written in C++ while the MAS framework employed is written in Java, a pipe based inter-process communication module was developed to bridge the MAS and modified ball detection system as explained in detail in Section 3.4. While the BDAs take the responsibility of communication with the ball detection system, MVCA controls those BDAs by providing commands such as GET, SLEEP, WAKE and PUT.

- The GET command is used for getting the ball detection results from BDAs.
- The SLEEP command is used for suspending the detection process for a specified time.
- The WAKE command is used for waking up from SLEEP and resume the detection.
- The PUT command is used for sending the result back to BDAs for multi-view correction.

Figure 5.7 shows the connections and information flows within the system. At the start, the MVCA instructs (GET) all BDAs to report the ball position as it has an ability to automatically detect the number of different types of agents which are currently running in the system. If the ball is visible in a BDA's view, it will return the ball position. Otherwise, it will state "No Ball" and hibernate. The 3D

position of the ball derived by a BDA is with respect to the principal point of one stereo camera as the origin. As the MAS system can have more than one BDA, the MVCA remaps the 3D ball position derived by each BDA into one real-world 3D ball position by using a common origin as presented in Chapter 3, Section 3.3.5.



BDA: Ball Detection Agent

MVCA: Multi-view Correction Agent

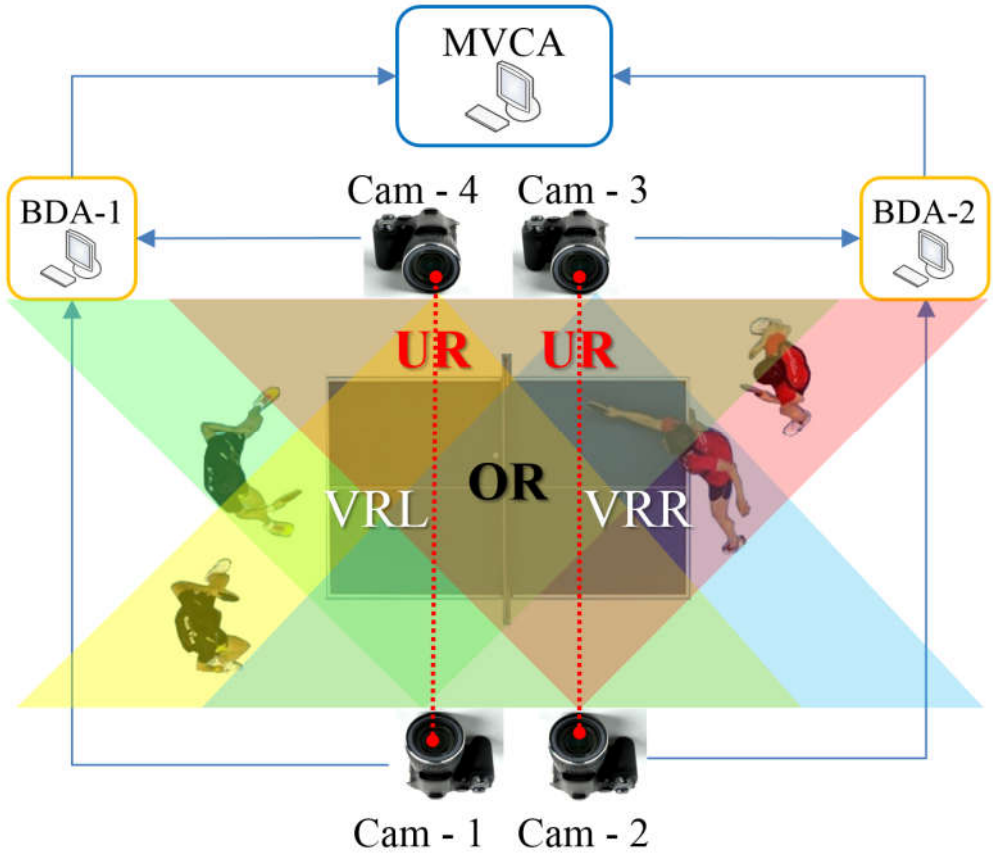
Figure 5.7: Architecture of the MAS

Since the MVCA maintains successive ball positions, it has an ability to predict the trajectory of the ball. In this way, the MVCA can check the consistency of the ball's 3D position from multiple BDAs and use this information to derive the most likely ball position (Multi-view correction). Whenever the MVCA receives ball positions from a different BDA, it compares the incoming data with its predicted data. If the different is higher than the running average error and the

detection score of that BDA is low, the MVCA can recognise that an incorrect ball location is given by a particular BDA. For that scenario, the MVCA sends back (PUT) the corrected screen position of the ball to the BDA that manages that view for correction. To be able to send back (PUT) the 3D ball position or wake up (WAKE) the relevant BDA, the MVCA also has an ability to convert the real-world 3D ball position to a related 3D position of each BDA. Figure 5.8 presents the multi-view cameras setup and different visible regions. Each opposite-facing pair of cameras is connected to a BDA. Each of the two BDAs detects the 2D screen position of the ball from each view, derives the 3D real world position of the ball and sends it to the MVCA for storage. The BDAs also provide the current frame number, the radius and the detection score of the current ball detection result to MVCA for further analysis. As the play region is monitored by multiple BDAs, the MVCA needs to know which regions of the play region are monitored by which BDAs. These 3D regions are defined based on the camera positions and inputted to MVCA during initialisation. Based on this knowledge, the MVCA can determine which visible region the current incoming ball position is located in and which type of command should be given to different BDAs such as GET, SLEEP, WAKE, PUT.

As mentioned in Section 3.3.5, when the position of the ball reaches the vertical plane that joins the principal points of the opposite facing cameras, the 3D position cannot be determined using the triangulation equations because the angles between the ball and the two cameras are zero. In fact, when the ball is near the plane, the calculated 3D position is erratic. As a result, a 3D region is defined as an Undetectable Region (UR) as shown in figure 5.8 with red dotted

lines. When the ball is crossing the UR, the MVCA extrapolates the 3D ball position based on the previous successful detection results. As each BDA only monitors a portion of the table, the MVCA selectively instructs which BDA to report the ball position and which to hibernate by determining which BDA(s) can view the ball. The MVCA will subsequently send the 3D ball position to other agents (not shown in Figure 5.8) for further umpiring purposes. The BDAs obtain the ball position through a pipe connection to the ball detection system which detects the ball from video feeds of the opposite facing cameras. The process continues until the ball appears in the Overlapping Region (OR) where the ball is visible by both camera pairs (the region near the net, the middle of the table as shown in Figure 5.8).



BDA-1: Ball Detection Agent 1

BDA-2: Ball Detection Agent 2

MVCA: Multi-view Correction Agent

UR: Undetectable Regions (Red dotted lines)

OR: Overlapping Region

VRL: Visible Regions Left

VRR: Visible Regions Right

Figure 5.8: Multi-view Cameras setup and different visible regions

While the ball is in the OR, the MVCA will receive the ball positions from different BDAs which are currently detecting the same ball at the same time. If the ball positions among BDAs are not consistent, the MVCA will choose the 3D ball position based on the previous successful detection history, predicted

trajectory and the confidence value of the different BDAs. As the ball is travelling from one end to the other end of the table, the ball will eventually disappear from the view of the BDA which initially observed the ball. When this happens, the MVCA will instruct that BDA to SLEEP, and the relevant BDA to wake up and report the ball position. The process continues until it reaches the end of the rally.

5.4 Experimental Results

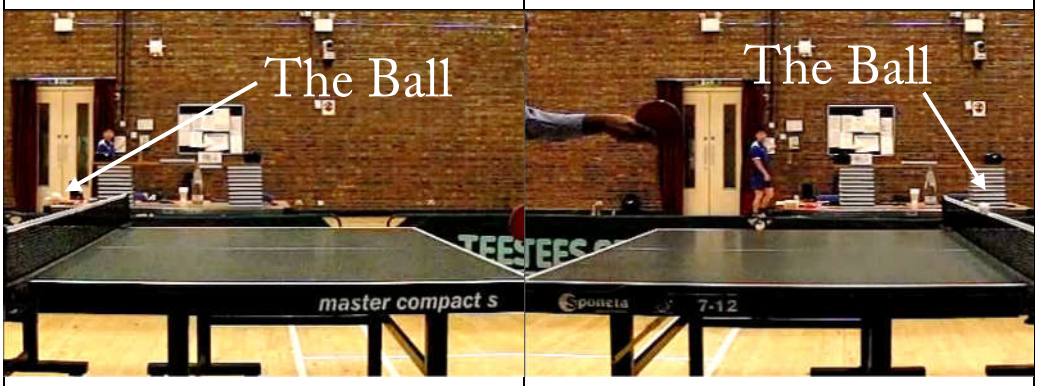
The system was tested with ten 4-view video sequences captured at a match scene. Each view of the sequences has a resolution of 512×384 pixels and was captured at 300 frames per second (fps). The choice of this resolution was limited by the low-cost entry-level high-speed cameras that are employed. Although the capturing rate is sufficiently high enough to detect the high-speed flying ball, the video is of low contrast and appears dark. Cyclic variation of illumination is also noticed due to the capture rate being much higher than the frequency (50 Hz) of the alternating current ceiling lights. As mentioned in 5.1, sequence 1 to 3 are tested in the Chapter 4 and the rest of sequence 4 to 10 are tested in this chapter. The detailed explanation of the characteristic of tested sequences can be found in Section 3.5. Each sequence contains different events of a typical table tennis rally will have and can confidently be used to test the system. As explained in Section 3.3, a checkerboard was carefully placed at various known positions during the calibration process to get a set of reference points. The real-world distance (cm) between each camera and those reference points was carefully measured and it is used for results comparison to assure that

the system can provide the right 2D to 3D reprojection results. The 3D derivation and evaluation methodology can be found in Chapter 3, Section 3.5. The detected 2D ball locations on each view are compared with the ball locations identified by human volunteers to evaluate the detection performance of the system. Three key quantitative parameters, the average RMSE, Detection rate and Processing time, have been used throughout this thesis as benchmarks to validate the performance results.

5.4.1 Sequence 4: Results Discussion

This four views sequence is composed of 3156 identified ball locations, taking 3 seconds to complete a rally. The chosen sequence consists of a complete table tennis rally in which the ball is coming from Visible Region Right (VRR), bounces on its own court, and crosses over the net. When the ball is out of VRR and entering in the scope of Visible Region Left (VRL), it is continuously detected by BDA1 until the rally is ended with a fault. The fault occurs with double bounces on the opponent's court after the ball hits the net and the net gets several vibrations. This sequence is selected to test the detection performance of the system during multiple motion and uneven lighting conditions. Figure 5.9 shows an example frame of four views in which the left (top and bottom) is captured by opposite facing cameras 1 and 4, the right (top and bottom) is captured by camera 2 and 3. This figure shows the challenging environment of the ball detection among confusing objects.

Camera 1 View at frame 560 | Camera 2 View at frame 560

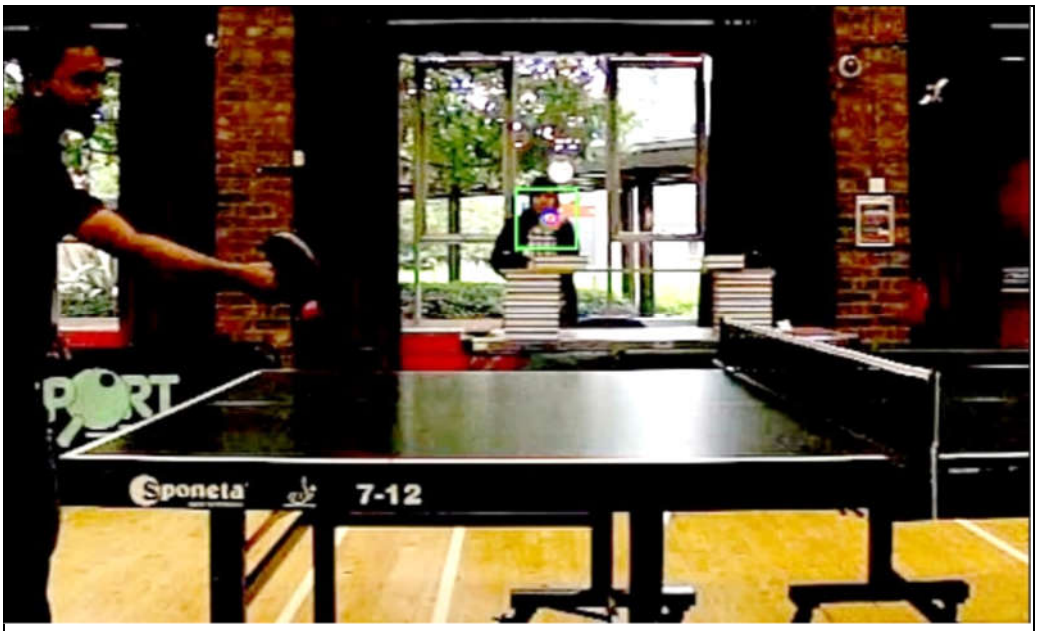


Camera four views at frame 560 | Camera 3 View at frame 560

Figure 5.9: Four views of sequence 4: The ball about to hit the net

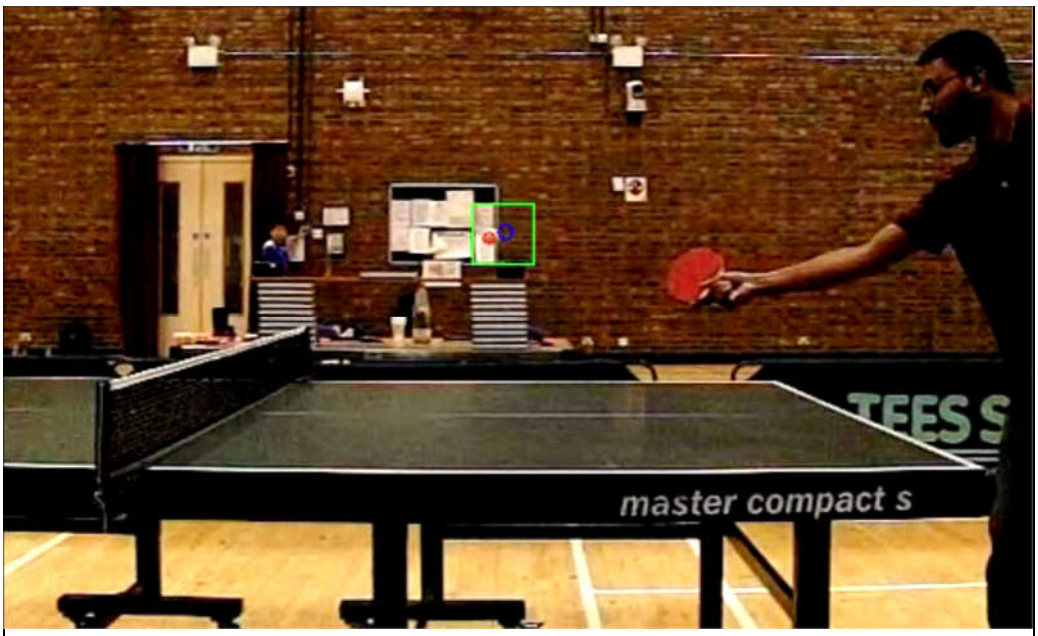
Figure 5.10 and 5.11 show the detection results of one example frame in which the ball is travelling against a similar colour background in visible region VRL. The visible region VRL is controlled by the BDA1 which is composed with the opposite-facing cameras Camera 1 and Camera 4. In these figures, the blue and red circles indicate the predicted and detected positions of the ball. The small letters (x, y) indicate the screen's coordinate (pixel) positions where the left top corner of the screen is (0,0) and r stands for the radius of the ball. As mentioned in Section 3.3, a common point (Top of the Net Pole in front of Camera 1 and Camera 2) was set as a NPO (0, 0, 0) to synchronise all the 3D values of each BDA. Therefore, the 3D (X, Y, Z) reprojection results of (-54, 24, 31) can be

translated as: the ball is currently located at -54 cm on the left side, 24 cm above and 31 cm depth, measured from the NPO after applying the correction using the error model.



Camera 1-Frame No. 282	x	y	r
Ground Truth (pixels)	273	179	3
System Result (pixels)	275	179	3
RMSE (2D)	2 pixels		
Detection Time	0.028 seconds		
3D (X, Y, Z) based on BDA1	(-54, 24, 31) cm		

Figure 5.10: Detected results of the ball at Frame No: 282 (Camera 1)



Camera 4-Frame No. 282	X	y	r
Ground Truth (pixels)	244	185	3
System Result (pixels)	243	185	3
RMSE (2D)	1 pixels		
Detection Time	0.028 seconds		
3D (X, Y, Z) based on BDA1	(-54, 24, 31) cm		

Figure 5.11: Detected results of the ball at Frame No: 282 (Camera 4)

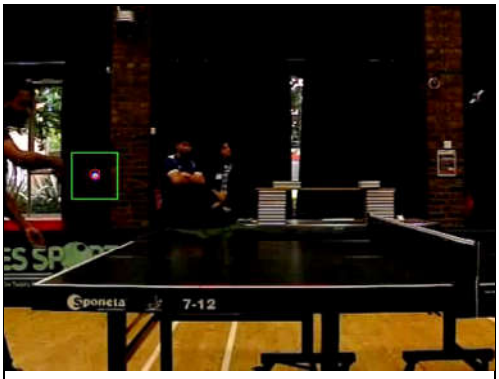
Table 5.1 shows the quantitative result of each camera in this sequence. To visualise the detail detection-results of the whole rally in an image, figures A.1 to A.4 in the Appendix show the 2D flight path comparison between the ground truth and the detected ball locations by BDA1 and BDA2. Although detected ball trajectories are similar to the ground truth, the detection errors occur at the place where the ball hit the net and when the image of the ball is blocked by the players, their bats or net. However, the system is able to recover and resume successful detections.

Table 5.1: Quantitative Result of each camera in sequence 4

	BDA1		BDA2	
Quantitative Result	Camera 1	Camera 4	Camera 2	Camera 3
Total Frame	876	876	702	702
Detection rate	92%	90%	83%	95%
Average radius of the ball	3.5 pixels			
Average RMSE (X, Y, R)	3.2 pixels	3.9 pixels	3.6 pixels	2.1 pixels
Average Processing time	0.027 seconds			

5.4.2 Sequence 5: Results Discussion:

This four views sequence is composed of 3282 identified ball locations and taking 3 seconds to complete a rally. As it is a complete rally, it consists of a service strike, the ball bouncing on the playing surface, the ball travelling back and forth across over the net, striking by the players, and an eventual foul. This represents a typical table tennis rally. The fault occurs when the ball drops under the table and disappears from all camera views. This is a challenging sequence because the system needs to detect the ball among surrounding objects which exhibit different motions. The figures 5.16 shows the detection results of frame 1 from the view of BDA1 which is composed of Camera 1 and Camera 4.



Camera 1-Frame No. 1	x	y	r
Ground Truth (pixels)	91	190	5
System Result (pixels)	**91**	**191**	**5**
RMSE (2D)	1 pixel		
Detection Time	0.028		
3D (X, Y, Z) based on BDA1	**(-152, 22, 145) cm**		

Camera 4-Frame No. 1	x	y	r
Ground Truth (pixels)	508	169	4
System Result (pixels)	**508**	**169**	**5**
RMSE (2D)	1 pixel		
Detection Time	0.028		
3D (X, Y, Z) based on BDA1	**(-152, 22, 145) cm**		

Figure 5.12: Example Results of BDA1 (Partially see the service)

In the middle of the play (starting from frame 648 till 723), the ball reaches out of Camera 2 cameras' views and disappears for more than 75 frames as shown in

figures 5.13 to 5.15. In this time, the system uses the information from Camera 3 and waits for the ball return to derive the 3D location of the ball. At the same time, the system expands the ROI (the green rectangle shown in Figure 5.13) at where the ball last appears to increase the chance of finding the ball. In this way, the system can successfully capture the returning ball at frame 722 and can continue the detection as shown in figures 5.14 in which the green circle indicates the expected position of the returning ball and the red circle shows the detected location of the ball. As multiple occlusions occur very frequently, this selected sequence 2 is chosen to test the robustness of the inter-view correction, MVCA and SDA.

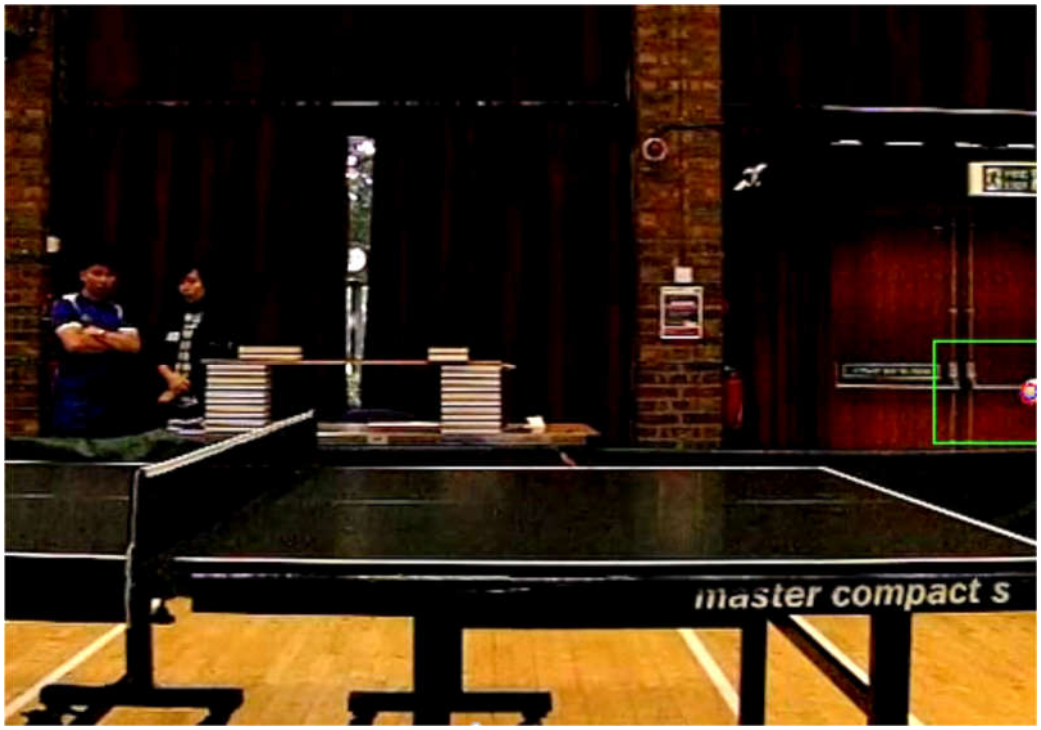


Figure 5.13: The Outgoing ball from the view of Camera 2 at frame 648

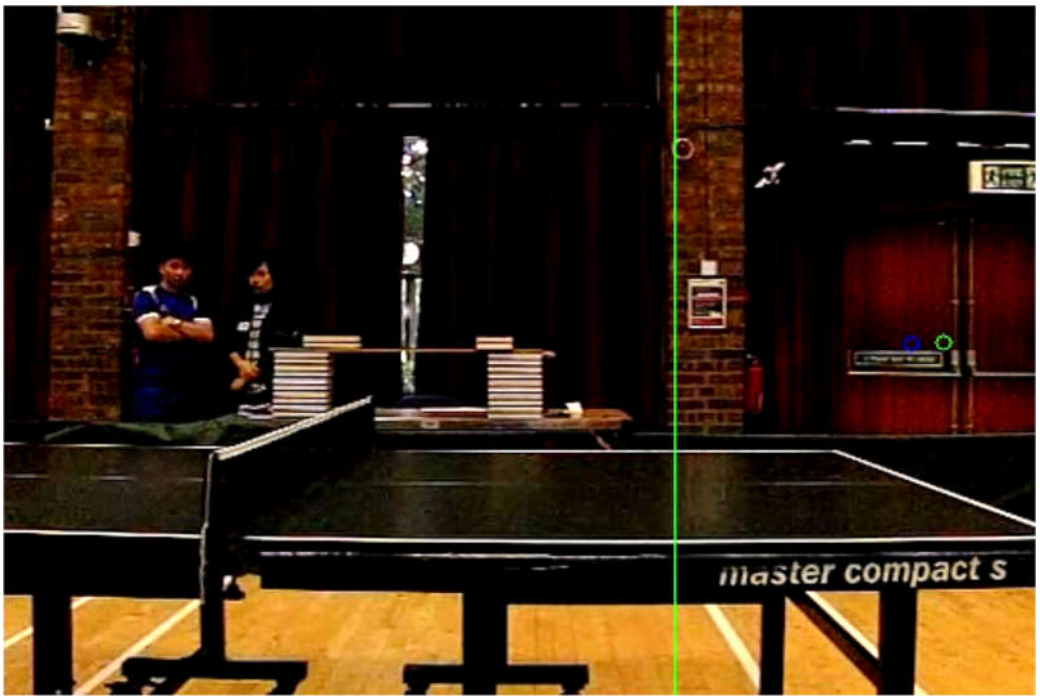


Figure 5.14: The ball disappears from the view of Camera 2

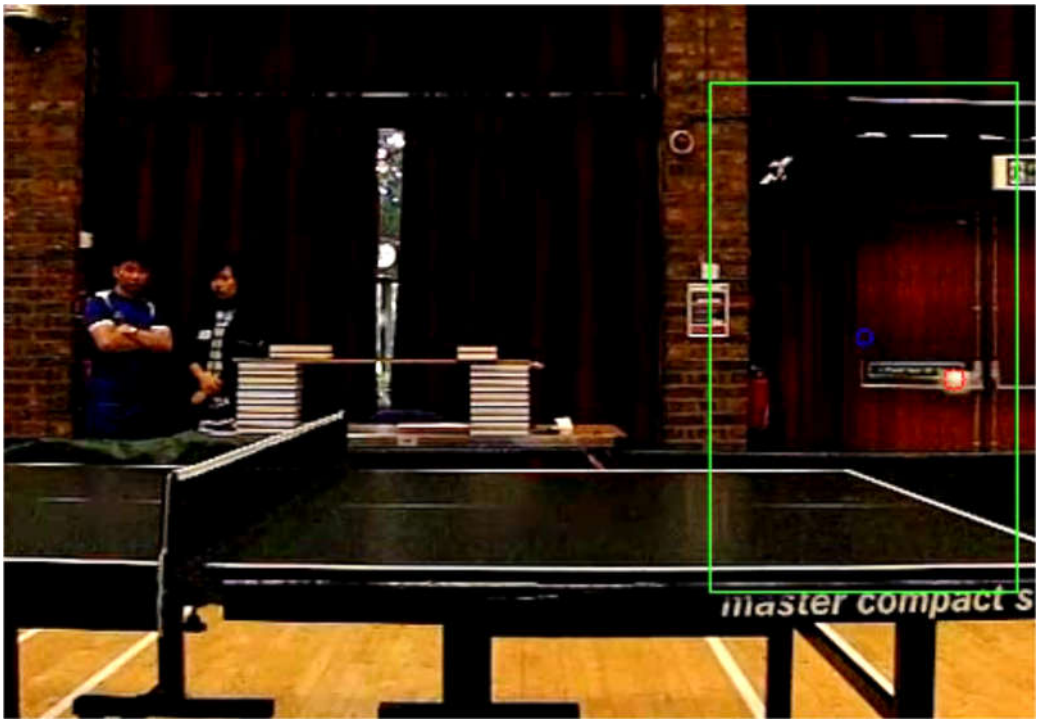
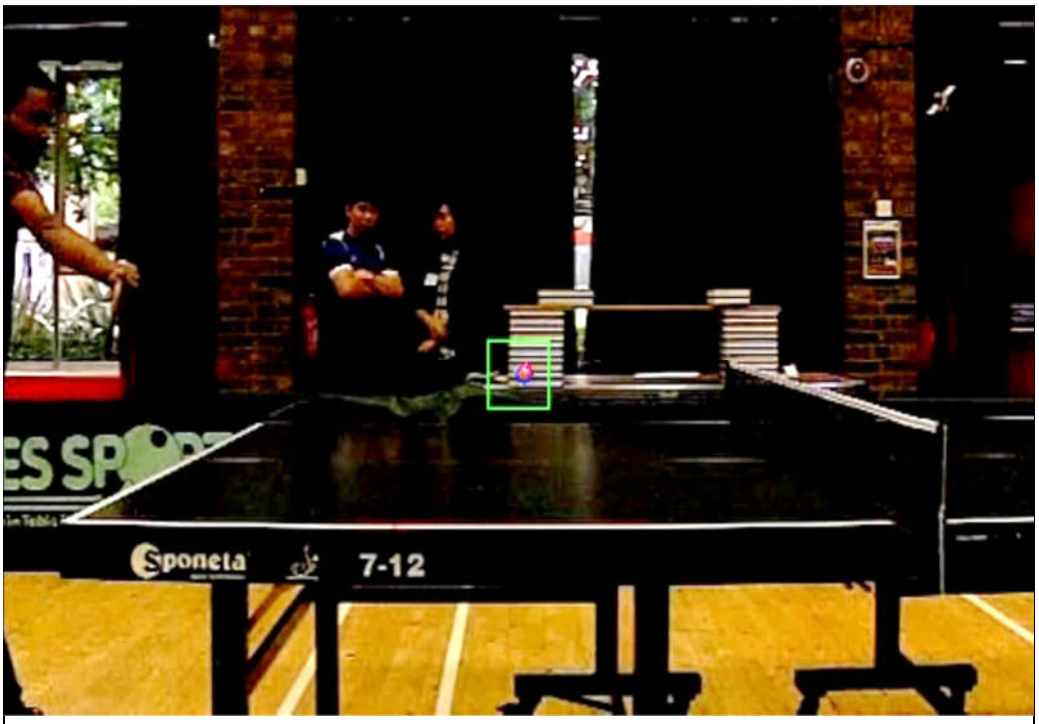


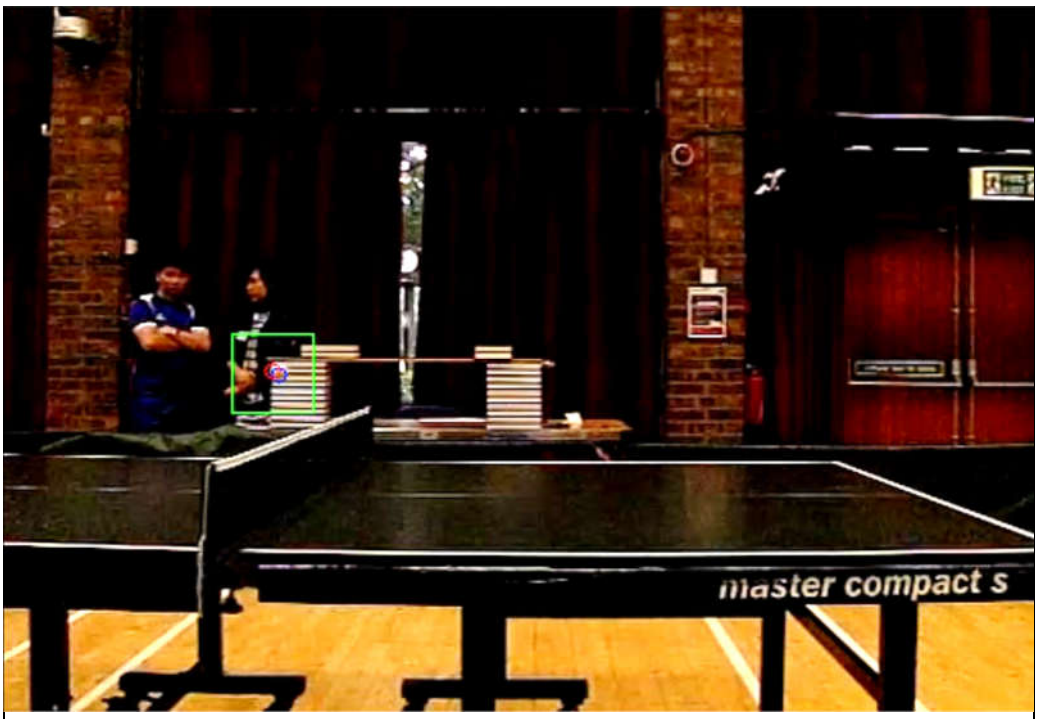
Figure 5.15: The returning ball after disappeared

Figure 5.16 to 5.19 show the views of the four cameras and the synchronous detection results at an example frame where the ball is crossing the OR and it is successfully detected by all four cameras.



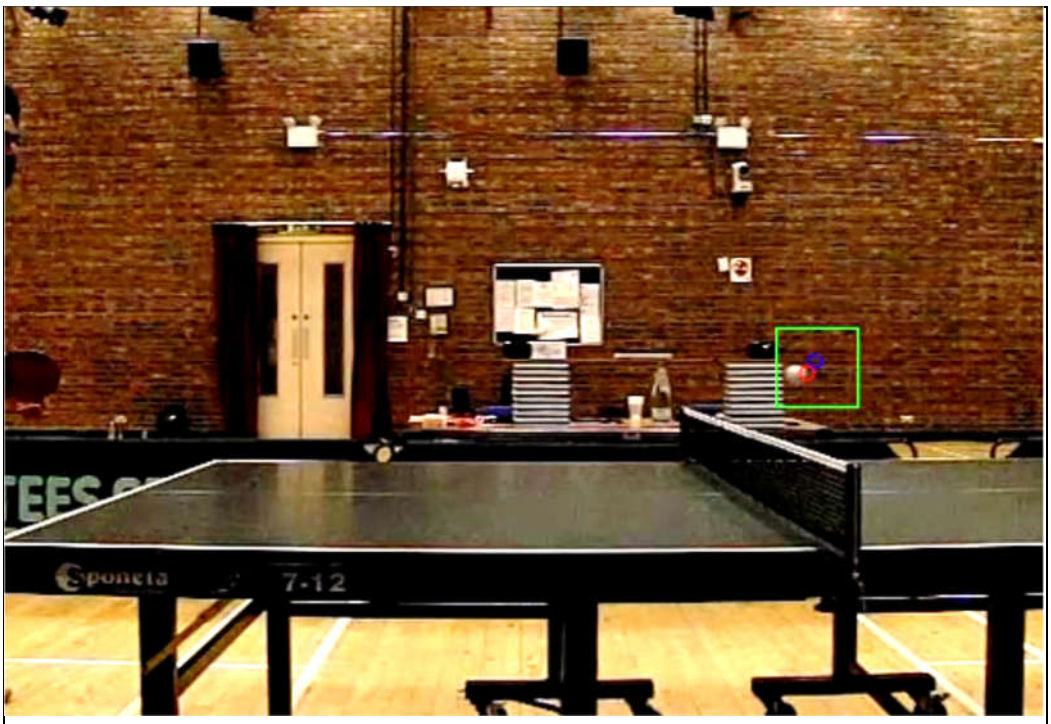
Camera 1-Frame No. 105	X	y	r
Ground Truth (pixels)	363	208	3
System Result (pixels)	363	207	3
RMSE (2D)	1.4 pixels		
Detection Time	0.028		
3D (X, Y, Z) from BDA1	(-10, 12, 102) cm		

Figure 5.16: Camera 1: Detected results at Frame No: 105



Camera 2 -Frame No. 105	X	y	r
Ground Truth (pixels)	135	209	3
System Result (pixels)	134	209	4
RMSE (2D)	1.4 pixels		
Detection Time	0.028		
3D (X, Y, Z) from BDA2	(-9, 12, 103) cm		

Figure 5.17: Camera 2: Detected results at Frame No: 105



Camera 3-Frame No. 105	X	y	r
Ground Truth (pixels)	394	211	3
System Result (pixels)	395	210	4
RMSE (2D)	1.7 pixels		
Detection Time	0.028		
3D (X, Y, Z) from BDA2	(-9, 12, 103) cm		

Figure 5.18: Camera 3: Detected results at Frame No: 105

Camera 4-Frame No. 105	x	y	r
Ground Truth (pixels)	135	212	3.2
System Result (pixels)	136	211	3
RMSE (2D)	1.4 pixels		
Detection Time	0.028		
3D (X, Y, Z) from BDA1	(-10, 12, 102) cm		

Figure 5.19: Camera 4: Detected results at Frame No: 105

In figure 5.16 to 5.19, the 3D (X, Y, Z) reprojection results of BDA1 is (-10, 12, 102) cm and BDA2 is (-9, 12, 103) cm respectively. As can be seen in these figures, BDA1 and BDA2 can produce slightly different 3D measurement results due to misalignments between the cameras, although the same ball was detected at some frames and at the same time. In this conflict scenario, the MVCA will decide the 3D position based on the predicted trajectory and the previous successful detection history of different BDAs. On this occasion, the MVCA

deems the result from BDA1 is more reliable and chooses (-10, 12, 102) cm as the ball location for this frame as presented in Table 5.2.

Table 5.2: MVCA's decided 3D result -Frame No. 105 from BDA1

Real world 3D Measurement	X	Y	Z
System Result	-10 cm	12 cm	102 cm

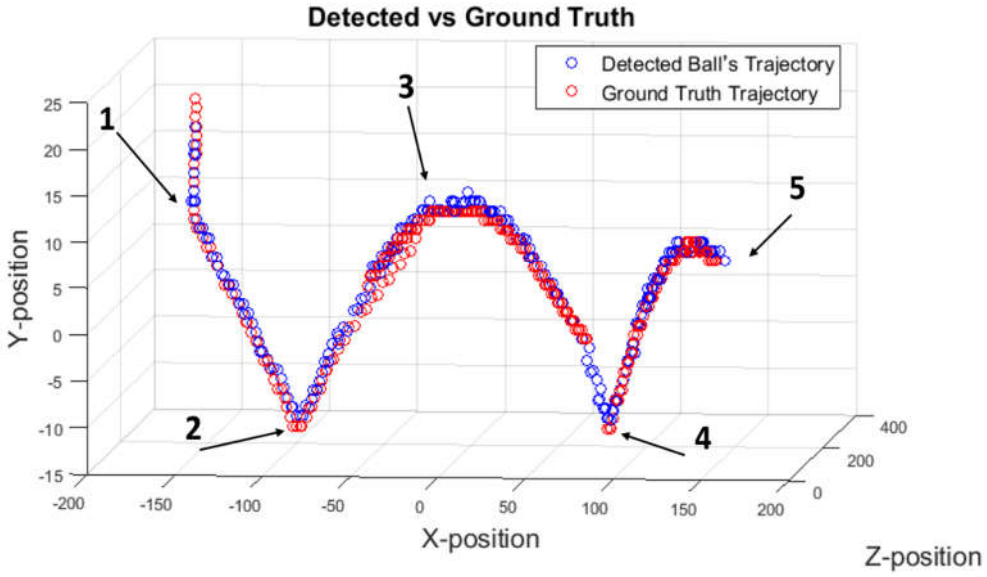


Figure 5.20: Server to Receiver: 3D Ball Trajectory travelling

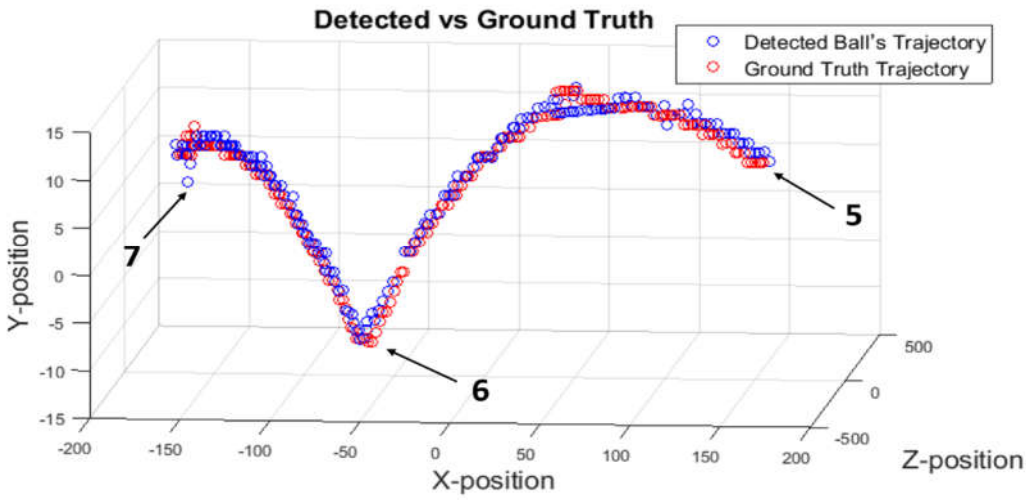


Figure 5.21: Receiver to Server: 3D Ball Trajectory travelling Labels:

1: Start of the service at Frame 1,
2: The ball bounces on the server side of the table
3: The ball crosses over the net
4: The ball bounces on the receiver side of the table
5: The ball is struck by the receiver
6: The ball bounces on the server side of the table
7: The rally ends.

To visualise the detail detection-result of whole rally in an image, figures A.5 to A.8 in the Appendix show the 2D flight path comparison between the ground truth and the detected ball locations by BDA1 and BDA2, and table 5.3 shows the quantitative result of each camera in this sequence.

Table 5.3: Quantitative Result of each camera in sequence 5

	BDA1		BDA2	
Quantitative Result	Camera 1	Camera 4	Camera 2	Camera 3
Total Frame	860	834	773	773
Detection rate	96%	98%	91%	96%
Average radius of the ball	3.5 pixels			
Average RMSE (X, Y, R)	3.2 pixels	3.9 pixels	2.9 pixels	2.6 pixels
Average Processing time	0.025 seconds			

5.4.3 Sequence 6: Results Discussion

This four views sequence is composed of 6780 identified ball locations taking 6 seconds to complete a rally. Among the ten sequences, this is the longest rally and is composed of 1800 frames. In the middle of the play (starting from frame 1325 till 1484), the ball reaches out of all cameras' views and disappears

for more than 150 frames. This is a challenging sequence because the system needs to be aware that the ball is temporarily out of all the cameras' scope and get ready to detect the returning ball. Since the rally is long, it challenges the system's ability of handling and synchronising the ball's position among different cameras. This sequence is selected to demonstrate the multi-view correction, and the prediction performance of the system during the ball disappearance from all cameras. After the ball travels back and forth across the playing surface multiple times, a fault occurs by the ball hitting the net and bouncing back on the player's own court. To visualise the detailed detection results of the whole rally in an image, figures A.9 to A.12 in the Appendix show the flight path history of the ball detected by BDA1 and BDA2, and table 5.4 shows the quantitative result of each camera in this sequence.

Table 5.4: Quantitative Result of each camera in sequence 6

	BDA1		BDA2	
Quantitative Result	Camera 1	Camera 4	Camera 2	Camera 3
Total Frame	1593	1593	1797	1797
Detection rate	94 %	88%	93 %	90 %
Average radius of the ball	3.5 pixels			
Average RMSE (X, Y, R)	1.1 pixels	1.1 pixels	3.5 pixels	2.5 pixels
Average Processing time	0.028 sec			

5.4.4 Sequence 7: Results Discussion

This four views sequence is composed of 1530 identified ball locations taking 2 seconds to complete a rally. In this sequence, the player is playing the ball in a diagonal direction and striking the ball with force. As the result, the ball

is travelling at high speed and the sizes of the ball in opposite cameras appear varies. When the ball is in high-motion, the image of the ball is distorted and its colour becomes blurred. This sequence is selected to test the detection performance of dynamic velocity changes, the ball's apparent shape variation, and colour merging with background objects. Instead of bouncing at the receiver side, this rally is ended when the ball goes beyond the table edge line without bouncing in the opponent's court after being struck by the opponent. To visualise the detail detection results of whole rally in an image, figures A.13 to A.16 in the Appendix shows the flight path history of the ball detected by BDA1 and BDA2, and table 5.5 shows the quantitative result of each camera in this sequence.

Table 5.5: Quantitative Result of each camera in sequence 7

	BDA1		BDA2	
Quantitative Result	Camera 1	Camera 4	Camera 2	Camera 3
Total Frame	325	325	440	440
Detection rate	99%	89%	87%	75%
Average radius of the ball	3.5 pixels			
Average RMSE (X, Y, R)	1.1 pixels	3.6 pixels	1.6 pixels	1.1 pixels
Average Processing time	0.027 sec			

5.4.5 Sequence 8: Results Discussion

This four views sequence is composed of 722 identified ball locations and takes 2.5 seconds to complete a rally. The system was intended to test for identifying different types of fault such as faults due to multiple bounces, faults due to a return not bouncing on the right side of the table and faults due to the

ball hitting the floor. In this sequence, the receiver misses the ball after being struck by an opponent. The ball passed over beyond the table edge line and it drops under the table. To visualise the detail detection results of whole rally in an image, figures A.17 to A.20 in the Appendix show the flight path history of the ball detected by BDA1 and BDA2, and table 5.6 shows the quantitative result of each camera in this sequence.

Table 5.6: Quantitative Result of each camera in sequence 8

	BDA1		BDA2	
Quantitative Result	Camera 1	Camera 4	Camera 2	Camera 3
Total Frame	245	245	116	116
Detection rate	98%	99%	97%	79%
Average radius of the ball	3.5 pixels			
Average RMSE (X, Y, R)	2.6 pixels	2.7 pixels	2.1 pixels	3.2 pixels
Average Processing time	0.021 sec			

As can be seen if figure 5.39, the system can successfully detect the ball bounce on the playing surface which is very close with the server end line. However, the detection can be lost when the image of the ball is dissolved in colour match background for several frames as shown in 5.40. However, the system can use the information from the opposite facing camera and continue to derive the 3D information by prediction. That predicted position will be used for umpiring.

5.4.6 Sequence 9: Results Discussion

This four views sequence is composed of 1988 identified ball locations and takes 2 seconds to complete a rally. In this sequence, the ball progresses from being served until the rally is ended by the ball touching the corner of the table and dropping down to the floor. Detecting the service part is challenging as there are several occlusions. To visualise the detail detection results of whole rally in an image, figures A.21 to A.24 in the Appendix show the flight path history of the ball detected by BDA1 and BDA2. As can be seen in figures, several detection errors occur at the place where the image of the ball is blocked by the player's hand and his bat. However, the system can detect the peak point of the ball, which is the necessary information to measure the ball rise for umpiring. Another challenging point in this sequence is detecting whether the ball touches the corner of the table or not. The system can identify the corner touches based on the detected location and the rate of change of velocity of the ball. The following table 5.7 shows the quantitative result of each camera in this sequence.

Table 5.7: Quantitative Result of each camera in sequence 9

	BDA1		BDA2	
Quantitative Result	Camera 1	Camera 4	Camera 2	Camera 3
Total Frame	565	565	429	429
Detection rate	84%	79%	98%	89%
Average radius of the ball	3.5 pixels			
Average RMSE (X, Y, R)	3.5 pixels	3 pixels	1.1 pixels	2.7 pixels
Average Processing time	0.039 sec			

5.4.7 Sequence 10: Results Discussion

This four views sequence is composed of 4492 identified ball locations and takes 4 seconds to complete a rally. Among the ten sequences, this is the brightest one with a lot of light reflections. Since all the windows are wide open, it has multiple moving background objects which can be confused and indistinguishable with the ball. In this sequence, the ball strongly hits the net while it is crossing the receiver court to the server and the net gets several vibrations. This sequence is selected to demonstrate the detection performance of the system during multiple motion allied with object blurring and colour deviation impact by the uneven lighting of the scene. To visualise the detail detection results of whole rally in an image, figures A.25 to A.28 in the Appendix show the flight path history of the ball detected by BDA1 and BDA2, and table 5.8 shows the quantitative result of each camera in this sequence.

Table 5.8: Quantitative Result of each camera in sequence 10

	BDA1		BDA2	
Quantitative Result	Camera 1	Camera 4	Camera 2	Camera 3
Total Frame	565	565	429	429
Detection rate	84%	79%	98%	89%
Average radius of the ball	3.5 pixels			
Average RMSE (X, Y, R)	3.5 pixels	3 pixels	1.1 pixels	2.7 pixels
Average Processing time	0.039 sec			

5.5 Result Comparison

The ball tracking system has been tested with a set of video sequences of a complete rally captured at a real match scene. Although the system overcame various detection challenges, some errors do occur where the ball hits the net or when the image of the ball is blocked by the players' bats or net. Errors occur especially at a region where the position of the ball is near the plane that joins the principal points of the opposite facing cameras. This means the 3D positions of the ball estimated by the second-order equation of motion can sometimes be inaccurate. Nevertheless, successful detections resume shortly after the ball passes that region. The detected ball trajectories are similar to the ground truth. The table 5.9 presents the quantitative result comparison of each tested sequence and identifies the system performance based on the experimental results of all sequences.

Overall, the system detection rate is 94%. While the average radius of the ball in each frame is around 3.5 pixels, the system can detect the ball with RMSE 2.4 pixel (due to large detection errors in a few occasions). This contribute the actual 3D error of RMSE distance is less than 2 cm. If the RMS error between the detected location and the ground truth is less than the diameter of the ball, the detection is assumed to be correct and this provides enough information to umpire a rally. The results video of playable file can be found in the OU library's database (ordo.open.ac.uk, 2019).

Table 5.9: Quantitative Result of each tested sequence and result comparison

Quantitative Result	Sequence 1	Sequence 2	Sequence 3	Sequence 4	Sequence 5	Sequence 6	Sequence 7	Sequence 8	Sequence 9	Sequence 10
Total Frame	184	1160	800	3156	3240	6780	1530	722	1988	4492
Detection rate	100%	91%	96%	90%	96%	92%	87%	95%	87%	92%
Average radius of the ball in pixels	4.5	1.95	3.4	3.5	3.5	3.5	3.5	3.5	3.5	3.5
Average RMSE (X, Y, R) in pixels	1.38	1.9	1.6	3.2	3.2	2.1	1.9	2.7	2.6	2.5
Average Processing time (Seconds)	0.01	0.059	0.047	0.027	0.025	0.028	0.027	0.021	0.039	0.027
System Performance Based on 10 different Sequences										
System Detection Rate	94% (Based on 10 different sequences)									
System Average RMSE (X, Y, R)	2.4 pixels									
System Detection Time	0.028 seconds									
Processing time for 1 second video (300 fps)	8.4 seconds									

5.6 Summary

This chapter presents a MAS-based ball-tracking system, which is designed to be low-cost, portable and fit for umpiring purposes. A multi-view camera configuration was designed such that a minimum number of cameras were required, yet it was able to cover a large area and provide enough video clarity for tackling the detection challenges. Furthermore, a measurement error correction model was derived, and it can significantly reduce the detection errors. As the cameras do not need to be fixed to the ceiling, the system is portable, which is very important as most table tennis tournaments take place at multi-purpose sport venues where installation of fixed equipment is not permissible. Although the experiment took place on a single computer, the design of the tracking system enables agents to be executed on a network of computers. This can spread the workload over a number of computers and significantly improve the overall performance such that real-time tracking is achievable. The average time taken to detect the ball from a frame is 0.028 sec. If each agent of the MAS (such as BDA1, BDA2 and MVCA) is run on a separate computer, this time is expected to be reduced to an acceptable delay for making an umpiring decision. The agent-based design also enables the system to be scalable. For example, if a larger playing area needs to be covered, more cameras (and agents) can be added to the system without significantly changing the program.

Chapter 6

A Multi-Agent System for Umpiring (Contribution Chapter)

6.1 Introduction

This chapter presents the design and development of block 6 (Umpiring Rallies), part of the proposed framework which has presented in Chapter 1, figure 1.2. The aim of this research is to develop an intelligent system which can track the location of the ball from live video images and evaluate a rally according to the standardised table tennis rules. As table tennis is a fast sport, judging its rallies is a complex task and requires a lot of decisions. The demand for timely and accurate observations of a rally imposed by the table tennis rules is also very high. Some of the judgements described by the table tennis rules (Delano and L.F. (n.d.), 2018) are very challenging even for professionally trained umpires, because many observations are required within a very short period of time. A particular example is whether the ball hits the edge of the table or rises above the minimum height requirement during services. To achieve that, the analysis must be conducted very quickly, and produce reliable judgements rapidly and consistently.

While the ultimate goal is to build an automatic umpiring system, one important requirement of such a system is to accurately and rapidly track the location of the ball during a match. The results of ball location play a crucial

role in analysing rallies and can seriously impact the decision making. Especially, the main difficulties of tracking the ball in a match are that the view of the ball can be occluded or merged with the background and becomes very difficult to detect. Example images of these challenging detection situations are shown in Figure 6.1 in which the image of the ball is blurry, colour matched with nearby objects, and very difficult to detect after it gets struck by the player. To tackle these problems, the previous chapter 5 proposed a multi-view ball tracking system by integrating the developed ball detection algorithm in chapter 4 with the concept of a MAS. Based on the experimental results, employing multiple cameras has been very helpful in detecting objects which are occluded from one viewing angle and the detection performance has been improved. With the effective coordination of agents, the system has been scalable and larger amounts of data fed from multiple cameras can be processed in parallel. If a larger playing area needs to be covered, more cameras (and agents) can be added to the system without significantly changing the program.

To sum up, this chapter presents an extended development of a multi-agent system for evaluating table tennis rallies. The developed system is composed of multiple interacting computing agents, which can perform specific tasks and decide for themselves what they need to do in order to achieve their objectives. The remainder of the chapter is organised as follows: Section 2 briefly provides the laws of table tennis. It is followed by the detailed explanation of the proposed state machine for analysing table tennis rallies in Section 3. The architecture, the roles and responsibilities of each agent are presented in Section 4. Experimental results and performance comparison are

given to verify the effectiveness of the proposed system in Section 5. The system has been compared against the judgements of human umpires for both accuracy and rate of response.

Figure 6.1: A blurry ball in each consecutive frame

6.2 Summarised Table Tennis Laws

The following descriptions are summarised rules based on ITTF (ITTF, 2018). At the beginning of the play, service shall start with the ball resting freely on the open palm of the server's stationary free hand. The server shall then project the ball near vertically upwards, without imparting spin, so that it rises at least 16 cm after leaving the palm of the free hand and then falls without touching anything before being struck. As the ball is falling, the server shall strike it so that it touches first his or her court and then touches directly the receiver's court; in doubles, the ball shall touch successively the right half court of server and receiver. From the start of service until it is struck, the ball shall be above the level of the playing surface and behind the server's end line, and it shall not be hidden from the receiver by the server or his or her doubles partner or by anything they wear or carry.

As soon as the ball has been projected, the server's free arm and hand shall be removed from the space between the ball and the net. The space between the ball and the net is defined by the ball, the net and its indefinite upward extension. It is the responsibility of the player to serve so that the umpire can be satisfied that he or she complies with the requirements of the Laws, and either may decide that a service is incorrect. If the umpire is not sure about the legality of a service he or she may, on the first occasion in a match, interrupt play and warn the server; but any subsequent service by that player which is not clearly legal shall be considered incorrect. Exceptionally, the umpire may relax the requirements for a correct service where he or she is satisfied that

compliance is prevented by physical disability. Figure 6.2 demonstrates an example of the serve and the play of table tennis in which the server (in light blue) serves the ball and the receiver (in black) waits to return the ball.

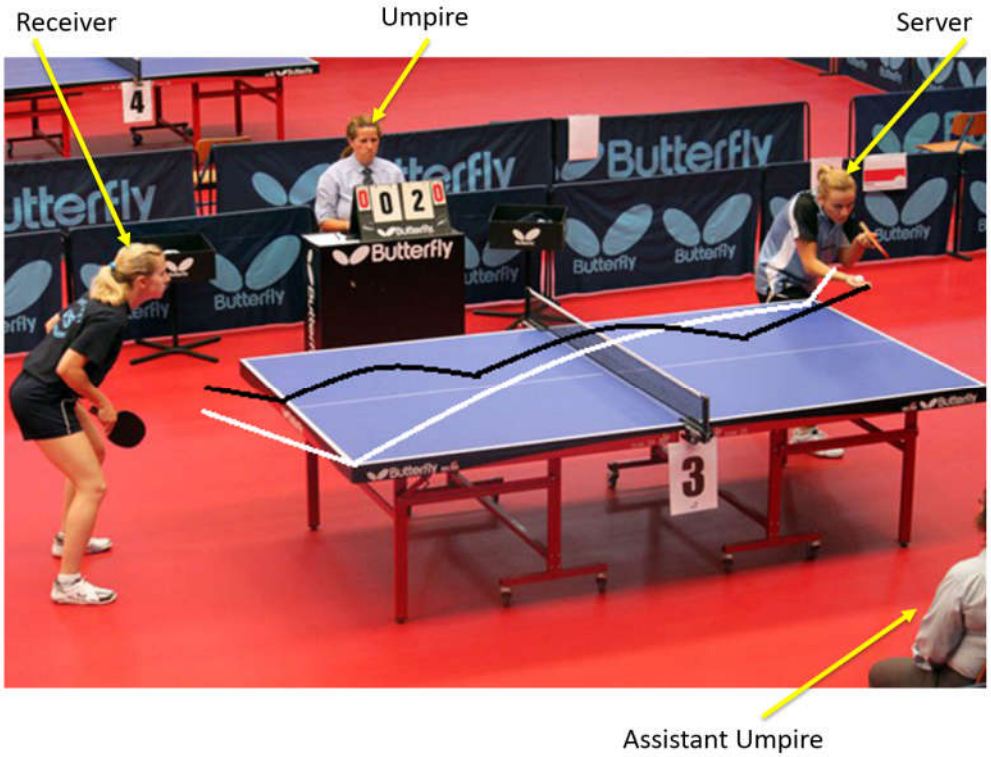


Figure 6.2: Demonstration of serve and play (Expert Table Tennis, 2013)

In the figure 6.2, both of the white and black lines are legal serves yet the white line represents a long serve in which the ball goes diagonally across the table from corner-to-corner, just catching the end line of the table and the black line represents a short serve, to the middle of the table, with a second bounce just clipping the end-line of the table. The ball, having been served or returned, shall be struck so that it touches the opponent's court, either directly or after touching the net assembly (ITTF, 2018).

During the service, if the ball touches the net and still bounces on the opponent's side of the table, the service must be replayed and is called a **Net Ball**. However, if the ball touches the net and does not bounce on the opponent's side of the table, the server loses the point. In singles, the server shall first make a service, the receiver shall then make a return and thereafter server and receiver alternately shall each make a return. A player strikes the ball if he or she touches it in play with his or her racket, held in the hand, or with his or her racket hand below the wrist. However, **Faulty Conditions** can occur when a player obstructs the ball which means:

- if a player, or anything the player wears or carries, touches the ball in play when it is above or travelling towards the playing surface;
- if the ball passes over the player court or beyond his or her end line without touching his or her court, after being struck by an opponent;
- if the ball, after being struck by an opponent, passes through the net or between the net and the net post or between the net and playing surface;
- if an opponent deliberately strikes the ball twice in succession.

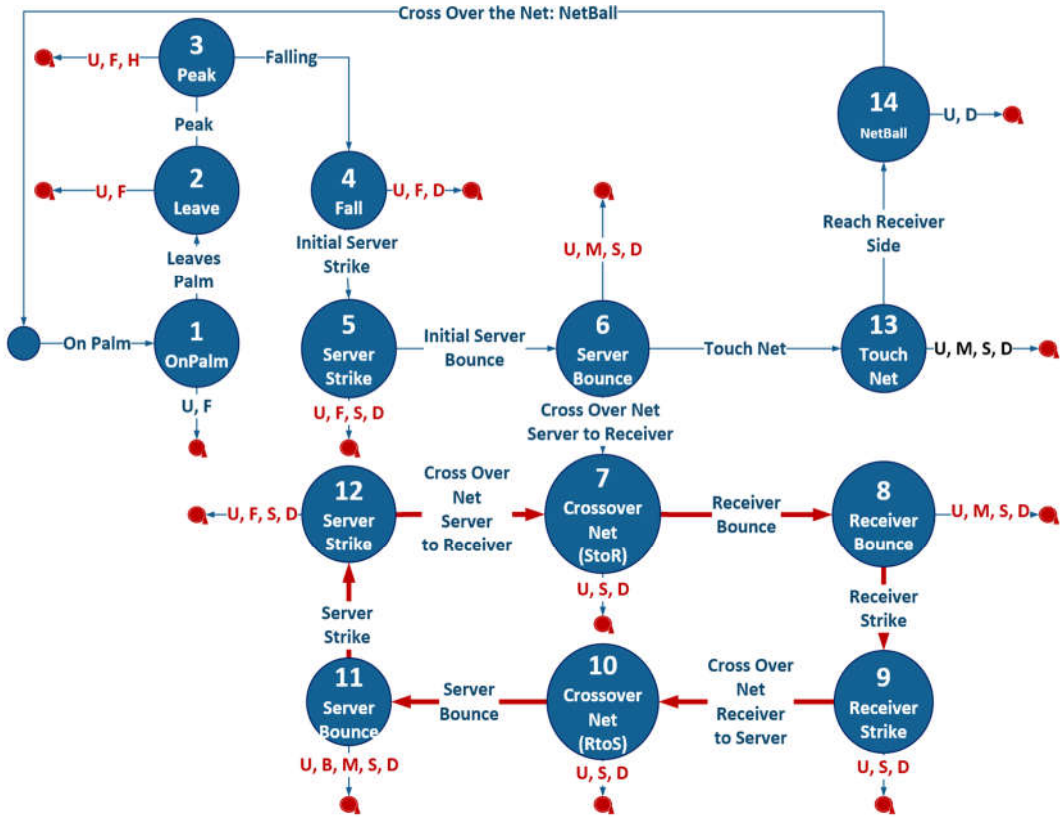
To this end, a purpose-built computer vision system could help evaluate these difficult observations. This chapter presents a prototype system which has been built and a variety of complex table tennis rallies were evaluated with the developed system. The developed system has an ability to declare when and what kind of fault occurs such as a fault due to double bounce, a fault due to a return not bouncing on the right side of the table, or a fault due to the ball dropping under table and hitting the floor etc.

6.3 Proposed State Machine for Analysing Rallies

To umpire the legality of the play, it is required to analyse table tennis rallies and implement some statistical rules for evaluation. Typically, a rally can be classified and identified as 14 possible states which are:

- State 1: Ball on palm
- State 2: Ball leaves palm and rises up
- State 3: Ball reaches its peak
- State 4: Ball is falling
- State 5: Ball is initially struck by the server
- State 6: Ball initially bounces on the server side
- State 7: Ball crosses over the net (server to receiver)
- State 8: Ball bounces on the receiver court
- State 9: Ball is struck by the receiver
- State 10: Ball crosses over the net (receiver to server)
- State 11: Ball bounces on the server side
- State 12: Ball is struck by the server
- State 13: Ball touches the net
- State 14: Net ball

Figure 6.3 illustrates the detected states of a table tennis rally in which the numbered blue circles indicate the different states and the red circles represent the faulty states.



U: Under the level of the playing surface

F: Front of server end line

H: Not high enough (distance between ball leave palm and peak is less than 16 cm)

M: Multiple Bounces

S: Skip a state

D: Disappear

Start of the rally

End of the rally or Exit with fault

Figure 6.3: Detected States of a table tennis rally

In the figure 6.3, the abbreviations **U, F, H, M, S,** and **D** stand for the distinct types of possible faults that can lead to the end of the rally.

- **U:** the rally is identified as faulty due to the ball being served below the level of the playing surface or dropped under the table.
- **F:** a faulty service because the ball is served in front of the server end line.
- **H:** the ball is not projected vertically upwards or not high enough (distance between ball leave palm and peak is less than 16 cm).
- **M:** the rally is faulty due to multiple bounces on the table.
- **S:** the rally skips a state or misses a step and exits as faulty.
- **D:** the ball disappears and didn't reappear within the acceptable time interval.

Among the six possible faults which are experimented upon in this chapter, **U, F, H,** and **D** can be found during a service and **U, M, S, and D** can happen at any stage for the rest of the rally. The blue arrow indicates that the ball is advancing from State to State and the red arrow illustrates that the ball is in play one state after another until the rally is ended by any type of fault. Each step of the evaluations is explained below in detail.

State 1: Ball on palm - First, the system must be able to distinguish whether the match is in play mode or not. Determining the start of a rally is challenging because many players have different pre-service actions, such as tossing the ball on their bats, table or the floor. Furthermore, there are a large variety of services the player can adopt. However, the server is required to place the ball on the open palm of his/her stationary free hand before the service starts. The ball usually stays on the open palm for one to a few seconds before the service starts. Based on this, as soon as the system can detect a short pause with the ball

stationary, it is used to identify the start of the service. From the start of service until the ball is struck, the ball should be above the level of the playing surface and behind the server's end line. If the ball is under the playing surface (**U**) or in front of server end line (**F**), the system will alert and exit with a fault as shown in figure 6.3. The server shall then project the ball near vertically upwards.

State 2: Ball leaves palm and rises up - When the system can detect the ball's rise, the state machine will switch to state 2 which is ball leaves palm and rises up. In this state, the ball should be above the level of the playing surface and behind the server's end line. If not, the system will identify whether the faulty state is caused by **U** which is under the level of the playing surface or **F**, in front of the server end line. There is also another possible faulty situation that the server may project the ball with force and the ball might go beyond the view. For this scenario, the system will try to trace back the returning ball by expanding the detection region as explained in chapter 4. If the ball does not return until the specified time, the system will alert and exit while declaring the fault **D**, Disappear.

State 3: Ball reaches its peak - As soon as the system can detect the peak of the ball's rise (a frame before falling), the state machine will switch to state 3. Then the system will make a necessary measurement such as the degree of the rise and the height. In this state, not only should the ball be above the level of the playing surface, it should also be behind the server's end line and rise at least 16 cm after leaving the palm of the free hand. Then, the ball should fall without touching anything before it is struck. If not, the system will identify whether the

faulty state is caused by any one of the four possible faults **U, F, H, D** and terminate.

State 4: Ball is falling - When the system can detect the ball's fall, the state machine will switch to state 4 which is ball falling. The faulty possibilities of this state are as the same as state 2. In this state, the ball should be above the level of the playing surface and behind the server's end line. If not, the system will identify whether the faulty state is caused by **U** which is under the level of the playing surface or **F**, front of server end line or **D**, disappear.

State 5: Ball is initially struck by the server - During the time when the ball is rising upward and falling downward without touching anything before it is struck, the displacement of the ball is nearly stable. As soon as the system can detect the sudden change of ball's velocity, it is assumed that the ball is struck by the server. From the start of service until it is struck, the ball should be above the level of the playing surface and behind the server's end line, and it should not be hidden from the receiver by the server. Otherwise, the system will identify whether the faulty state is caused by **U** which is under the level of the playing surface or **F**, front of server end line or **S**, miss to get struck, or **D**, disappear.

State 6: Ball initially bounces on the server side - According to the table tennis rules, the ball initially needs to touch server's court after it has been struck by him or her. In this state, the system needs to detect the ball's bounce. If the system can identify that the ball touches the level of playing surface and its velocity changes, the state machine will switch to state 6 which is server bounce.

In this state, the fault can be due to **U** which is the ball drops under the table or **M**, multiple bounces on its own court or **S**, doesn't bounce at all and crosses over the net or **D**, disappear from the view until the acceptable time limit.

State 7: Ball crosses over the net (server to receiver) - After the previous state 6, as soon as the system can detect the ball crosses over the net, it will switch to state 7 which is ball crosses over the net. Sometimes, the running ball may touch the net assembly. At that time, the system will announce the touch net and switch to state 13. Although the ball touches the net, if it can successfully reach the opponent's court, it is acceptable and either directly or after touching the net assembly (Net Ball).

State 8: Ball bounces on the receiver court - If the ball touches directly the receiver's court, after crossing over the net from server to receiver, the system switches to state 8, receiver bounce. In this state, the system needs to detect the ball's bounce. If the system can identify that the ball is touching the level of the playing surface and its velocity changes, the state machine will switch from state 7 to 8 which is server bounce. In this state, the fault can be due to **U** which is the ball drops under the table or **M**, multiple bounces on its own court or **S**, doesn't bounce at all or **D**, disappear from the view until the acceptable time limit.

State 9: Ball is struck by the receiver - After the previous state 8, as soon as the system can detect the sudden change of the ball's velocity and if the ball is travelling in a backward direction, it is assumed that the ball is being struck by the receiver. From the start of service until it is struck, the ball should be above

the level of the playing surface and behind the server's end line, and it shall not be hidden from the receiver by the server. In this state, the fault can be due to **U** which is the ball drops under the table or **M**, multiple bounces on receiver court or **S**, doesn't get struck at all or **D**, disappear from the view until the acceptable time limit.

State 10: Ball crosses over the net (receiver to server) - After the previous state 9, as soon as the system can detect the ball crosses over the net, it will switch to state 10 which is ball crosses over the net. Like previous states, the faulty state can be caused by either one of the three possible faults such as **U, S, D** and terminate.

State 11: Ball bounces on the server side - After the previous state 10, as soon as the system can detect the ball bounce on the server side, the state machine will switch to state 11 which is server bounce. Although the name and condition of detecting the bounce are similar to state 6, the previous state identifier is different. State 6 will only happen once in a rally although state 11 will happen again and again until the end of the rally. However, the fault can happen the same as state 6 such as the rally is ended due to **U** which is the ball drops under the table or **M**, multiple bounces on the table or **S**, doesn't bounce at all or **D**, disappear from the view and didn't come back until the acceptable time interval.

State 12: Ball is struck by the server - After the previous state 11, as soon as the system can detect the ball is being struck and if the ball is travelling in a backward direction, it will switch to state 12, server strike. In this state, the same faulty possibilities can be due to **U**, **M**, **S** or **D.**

State 13: Ball touches the net - As the ball is falling at state 4, the server should strike at state 5 so that it touches first his or her court at state 6 and then touches directly the receiver's court state 6. However, if the ball touches the net and does not bounce on the opponent's side of the table, the server loses the point and the system identifies a fault.

State 14: Net ball - During the service, if the ball touches the net at state 13 and still bounces on the opponent's side of the table, the service must be replayed, and the system is switched to state 14 which is Net Ball.

6.4 Multi-Agent System for Umpiring Table Tennis Rallies

To achieve umpiring decisions within a short period of time, more agents have been developed to reinforce the previously developed multi-agent ball detection system. Instead of one agent struggling to perform many tasks such as detecting, trajectory constructing, analysing that constructed trajectory and identifying the state of the rally, the proposed system is composed of five different types of agents which have specific knowledge and can individually perform an assigned task. This approach simplifies the implementation of a multi-agent automatic umpiring system and improves the overall performance. To contribute to the development of an automatic umpiring system, the system is composed of the following agents:

- Ball Detection Agents (BDAs)
- Multi-view Correction Agent (MVCA)

- Trajectory Construction Agent (TCA)
- Feature Detection Agent (FDA)
- State Detection Agent (SDA)

Ball Detection Agent (BDA): The BDA takes the responsibility of detecting and tracking the ball and determining the 3D location of the ball from a stereo view. It detects the 2D screen position of the ball from each view, derives the 3D real-world position of the ball and sends it to the MVCA. BDA also provides the current frame number, the radius and the detection score of the current ball detection result to MVCA for further analysis. As the system is developed as scalable, there can be more than one BDA according to the requirement and facility such as available number of cameras. If there are more cameras, the system can have more BDAs.

Multi-View Correction Agent (MVCA): The MVCA controls the BDAs by issuing commands such as **GET, WAKE, PUT** and **SLEEP** as explained detail in Chapter 5. At the start, the MVCA instructs all BDAs to report the ball position as it has an ability to automatically detect how many different types of agents are currently running in the system. If the ball is visible in a BDA's view, it will return the ball position. Otherwise, it will state "No Ball" and hibernate. Since the MVCA maintains successive ball positions, it has an ability to predict the trajectory of the ball for each BDA. In this way, the MVCA can check the consistency of the ball's 3D position from multiple BDAs and use this information to derive the most likely ball position. Whenever the MVCA receives ball positions from a different BDA, it compares the incoming data

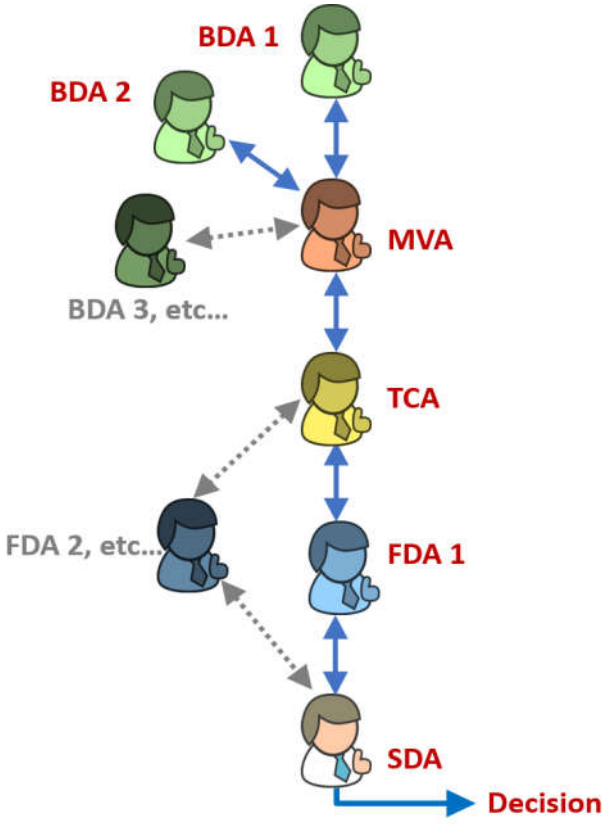
with its predicted data. If the gap is higher than the average error and the detection score of that BDA's detection is low, the MVCA can recognise that an incorrect ball location is given by a view. For that scenario, the MVCA sends back the corrected screen position of the ball to the BDA that manages that view for correction. After that, the MVCA sends the corrected ball positions from all BDAs to the TCA.

Trajectory Construction Agent (TCA): As soon as the TCA receives the corrected ball position from the BDAs via the MVCA, it constructs the joint trajectory. While the ball is in an overlapping area, the TCA will receive the duplicated ball positions from different BDAs which are currently detecting the same ball at the same time. If the ball positions among BDAs are not synchronised, the TCA will choose the most accurate 3D position based on the history of the detection performance, predicted trajectory and the detection score of the different BDAs. Like the MVCA which has an ability to predict the trajectory of the ball for each BDA, the TCA also has a predictor that estimates the whole trajectory by combining the results from all BDAs. In case any BDAs cannot detect the ball for some reason, the TCA can provide the approximate 3D ball location based on the knowledge of previous successful detection results. To prevent any misalignment errors that can occur while the ball is crossing from one visible region to another, the TCA smoothens out the whole trajectory based on the running average error before sending the results to the FDA for further analysis.

Feature Detection Agent (FDA): The FDA detects the trigger for changing one state to another. To understand the pattern of the ball's trajectory, the FDA analyses all the coordinates of the ball's 3D position. It considers, for example, whether the horizontal running displacement (X) or vertical height (Y) or the depth (Z) are increasing or decreasing, a sudden change of velocity or no movement at all, or the direction of the movement. Moreover, the FDA also detects the trigger of whether the ball gets struck, bounces on the table, crosses over the net, or touches the net. In terms of evaluating the correct play, the FDA also identifies whether the height of the ball is above or below or touching the table. Moreover, the FDA analyses whether the ball is travelling within or outside of the playing surface. In this way, the FDA can provide to the SDA knowledge of what kind of features are happening and where and when they occur.

State Detection Agent (SDA): The SDA controls the state machine and update the current state of the rally by analysing the features provided from the FDA. The SDA decides the current state, the previous state of the ball, what kind of features are currently and previously happening in the rally, and whether the rally is legal or not. To correctly identify the current state of the rally, the SDA learns the behaviour of the play and clarifies whether the current play is right or wrong by detecting every possible fault in each state such as whether the ball is served behind the server end-line or bounces on the correct ends of the table and so on. Figures 6.4 and 6.5 summarise the developed multi-agents and their functions, in which the arrow indicates the interaction between agents. BADs are used for detecting, tracking and reprojecting the ball 2D measurement into

3D. The MVA is developed for correcting errors using inter-view information and central control of BDAs. The TCA is mainly used for constructing the joint 3D trajectory and predicting the whole trajectory. The FDAs detect triggers for each state change. The SDA analyses features of the 3D trajectory and identifies each state change as well as faults. Depending on the system requirement, varying number of instances of all these types of agents can be added to reinforce the performance.



Where:

BDA1, BDA2 = Ball Detection Agents

MVA = Multiview Managing Agent

TCA = Trajectory Construction Agent

FDA = Feature Detection Agent

SDA = State Detection Agent

Figure 6.4: Relationship between Multi-agents

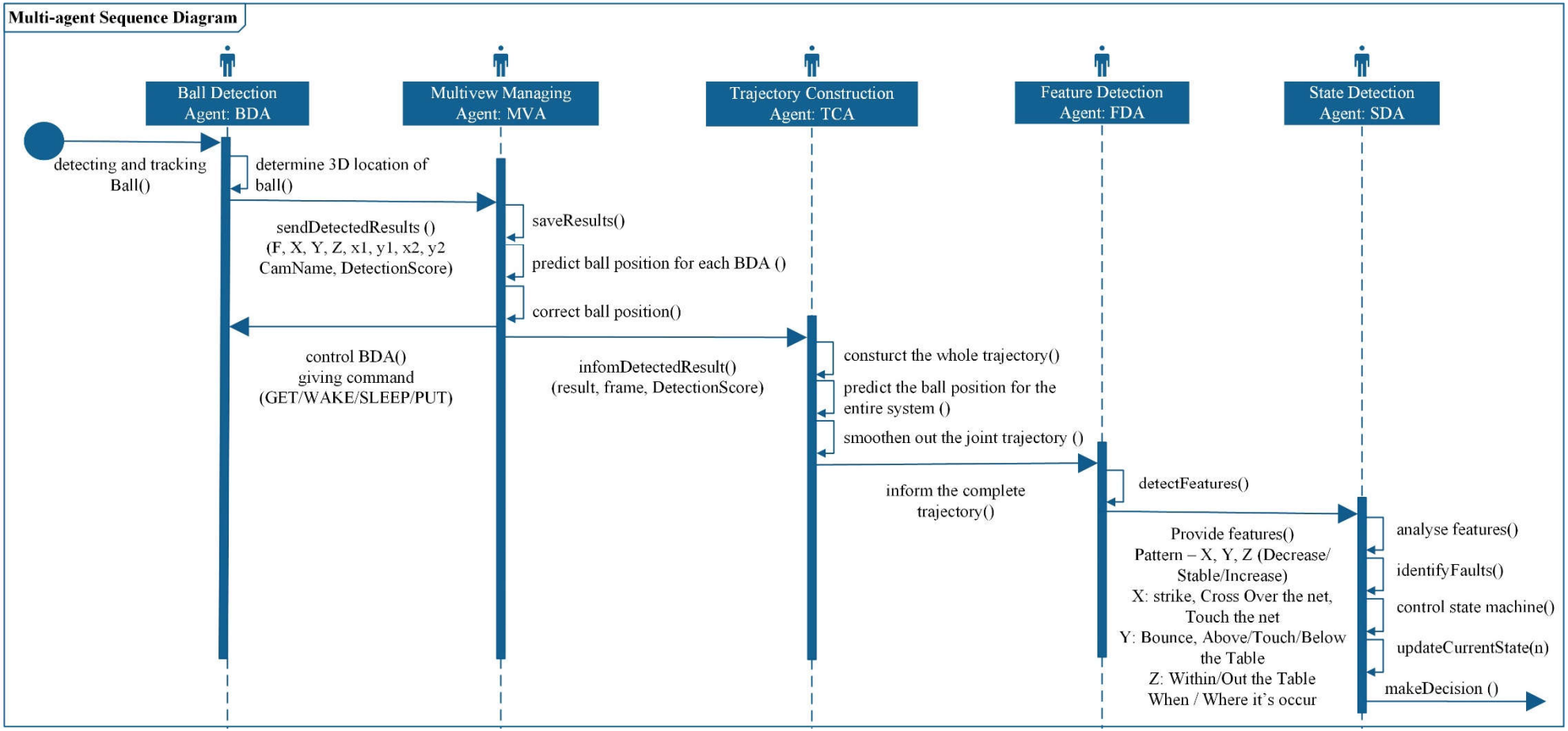


Figure 6.5: Multi-agents and their Responsibility

As specialised agents concentrate on performing their own task, many observations can be made in parallel. This approach can spread the workload over a number of agents and significantly improve the overall performance. Besides the agent abstraction, each agent can inquire the relative positions and orientations of all other agents performing tasks in its field of view. The system provides peer-to-peer agent communication based on the asynchronous message passing paradigm and the automatic agent discovery mechanism. Not only can these agents act autonomously, they also have a capability of interacting with other agents and utilising the results of one agent for another. Incorporating multiple agents and working together, the system can assist a human umpire by continuously providing the precise ball location in any possible situation, analysing the trajectory of a rally and determining the legality of the play. If necessary, the system can be expanded by increasing the number of agents to perform more sophisticated tasks. The developed system can be distributed across machines by moving agents from one machine to another when required.

6.5 Results and Discussion

The system was tested with several 4-view video sequences captured at a match scene. Each view of sequences has a resolution of 512×384 pixels and was captured at 300 frames per second (fps). As the system is primarily designed to aid umpires, the video was taken at a position and an angle similar to the umpire's perspective. This chapter provides an example of seven tested sequences which comprised different conditions of complete table tennis rallies

in which the different states of each rally are identified. A summary of the key features of the tested sequences is shown in Table 6.1 below.

Table 6.1: Summarisation of the tested sequences' characteristics

Sequence	Cover the state	Characteristic	End the rally due to
4	State 5 to 13	- Hits the net - Cross over the net	M: Double Bounces
5	State 1 to 12	- Partially see the service - Drop down the table - Disappear from all views	D: Disappear
6	State 3 to 12	- The longest rally - Double bounces	M: Multiple Bounces
7	State 6 to 12	- Without bouncing at the receiver side - It goes over and disappeared	O: Over Edge line D: Disappear
8	State 5 to 8	- Receiver misses the ball	S: Skip the state
9	State 1 to 10	- Partially see the service - The ball touches the corner of the table - Drop down the table and - Disappear from all views	U: Under table
10	State 4 to 11	- Very bright - The most complicated background - The ball hits the net - The ball crosses over the net and has multiple bounces on the table	M: Multiple Bounces

6.5.1 Sequence 4: Results Discussion

The chosen sequence consists of a complete table tennis rally in which the ball comes from VA2, bounces on its own court, and crosses over the net. When the ball is out of VA2 and entering in the scope of VA1, it was continuously detected by BDA1. The rally is ended by hitting the net with multiple bounces on the opponent's court. As can be seen in the figure 6.5, the whole trajectory is constructed based on the results of BDA1 (PC1) and BDA2 (PC2). The figure 6.6 shows the results of a multi-agent system for sequence 4 in which the ball is travelling from state 5 to state 12. The table 6.2 and the figure 6.7 present the comparison of state change frame. The system can identify every state correctly with the detailed 3D ball detection results with the average of 2 frames delay.

Table 6.2: Sequence 4 - State Change frame comparison

State	Ground Truth Frame No	Detected Frame No	Frame Delay	Average Frame Delay
5	1	1	0	2
6	45	45	0	
7	88	88	0	
8	188	192	4	
9	236	242	6	
10	316	316	0	
11	374	379	5	
12	463	469	6	
7	550	550	0	
8	628	628	0	
12	632	632	0	

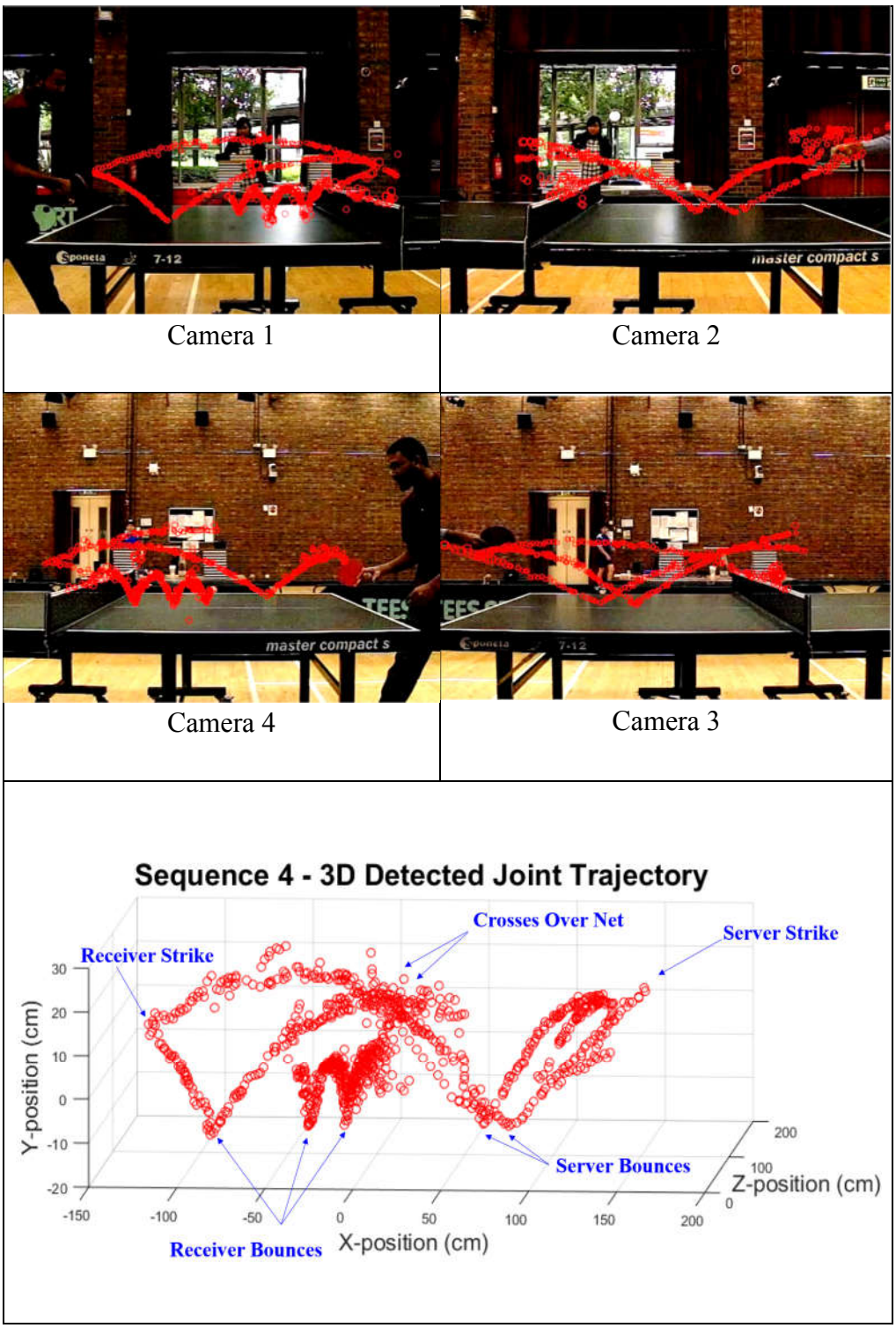


Figure 6.6: Sequence 4 - 3D Detected Joint Trajectory

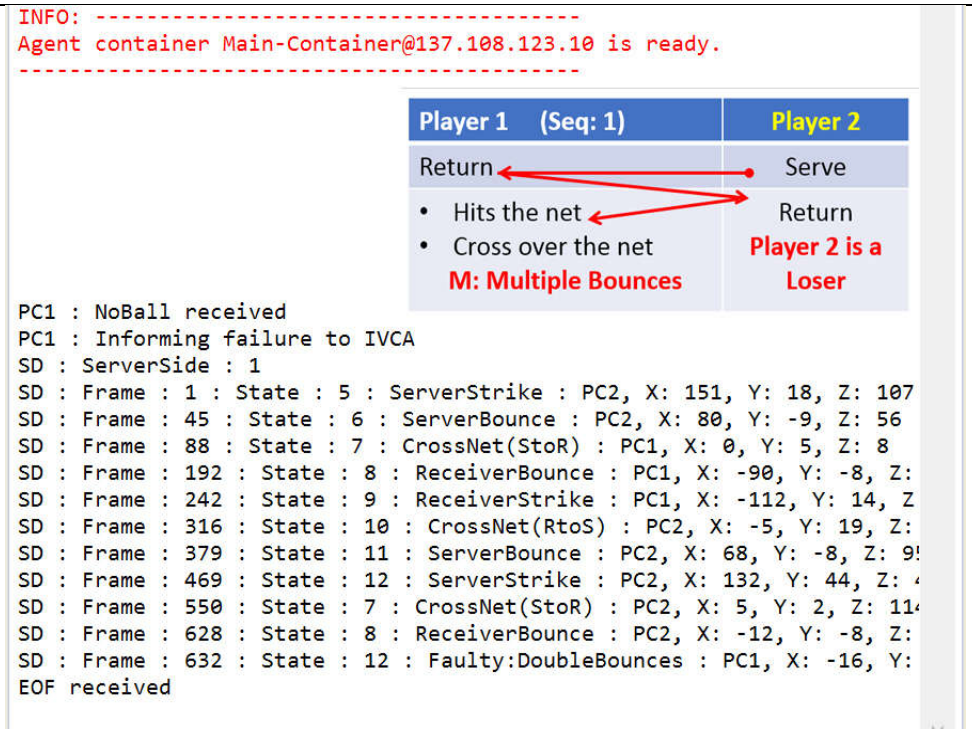


Figure 6.7: Sequence 4 - Results of multi-agent system

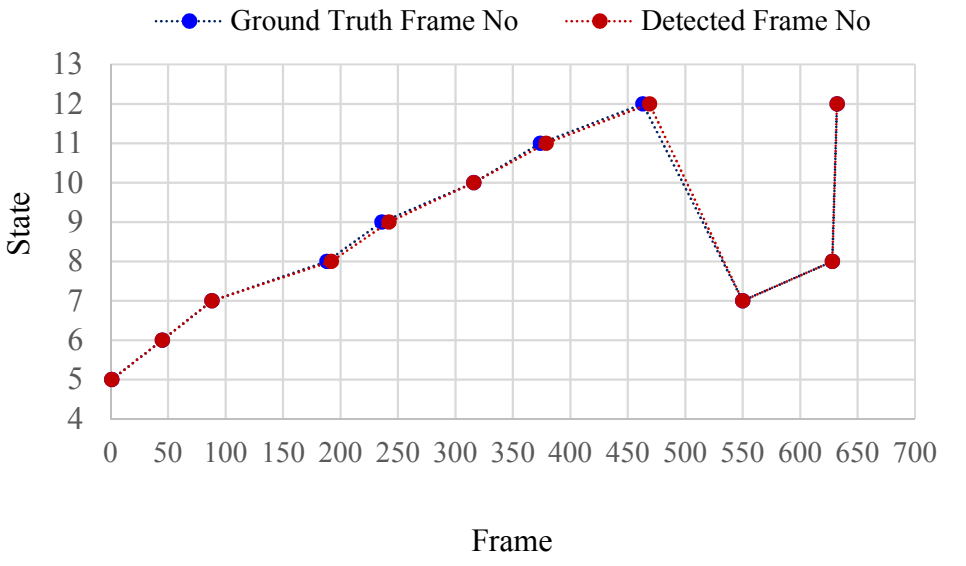


Figure 6.8: Sequence 4 - State Change Frame Comparison

6.5.2 Sequence 5: Results Discussion

This sequence consists of a complete table tennis rally in which the ball is served from VA1, is struck by the server, bounces on its own court, and crosses over the net. When the ball was out of VA1 and entering in the scope of VA2, it was continuously detected by BDA2 and so on until the rally was ended by the ball dropping under the table and disappearing from the view of all BDAs over the acceptable time interval and with the fault of **D:** Disappear. As can be seen in the figure 6.8, the whole trajectory is constructed based on the results of BDA1 (PC1) and BDA2 (PC2). The figure 6.9 provide the results of a multi-agent system for sequence 5. The table 6.3 and the figure 6.10 present the comparison of state change frame. The system can identify every state correctly with the detailed 3D ball detection results with the average of 4 frames delay.

Table 6.3: Sequence 5 - State Change frame comparison

State	Ground Truth Frame No	Detected Frame No	Frame Delay	Average Frame Delay
4	5	5	0	4
5	20	20	0	
6	50	65	15	
7	119	119	0	
8	186	187	1	
9	270	277	7	
10	343	343	0	
11	373	386	13	
12	448	456	8	
7	545	545	0	
8	603	606	3	
9	704	711	7	
10	769	769	0	
11	798	798	0	

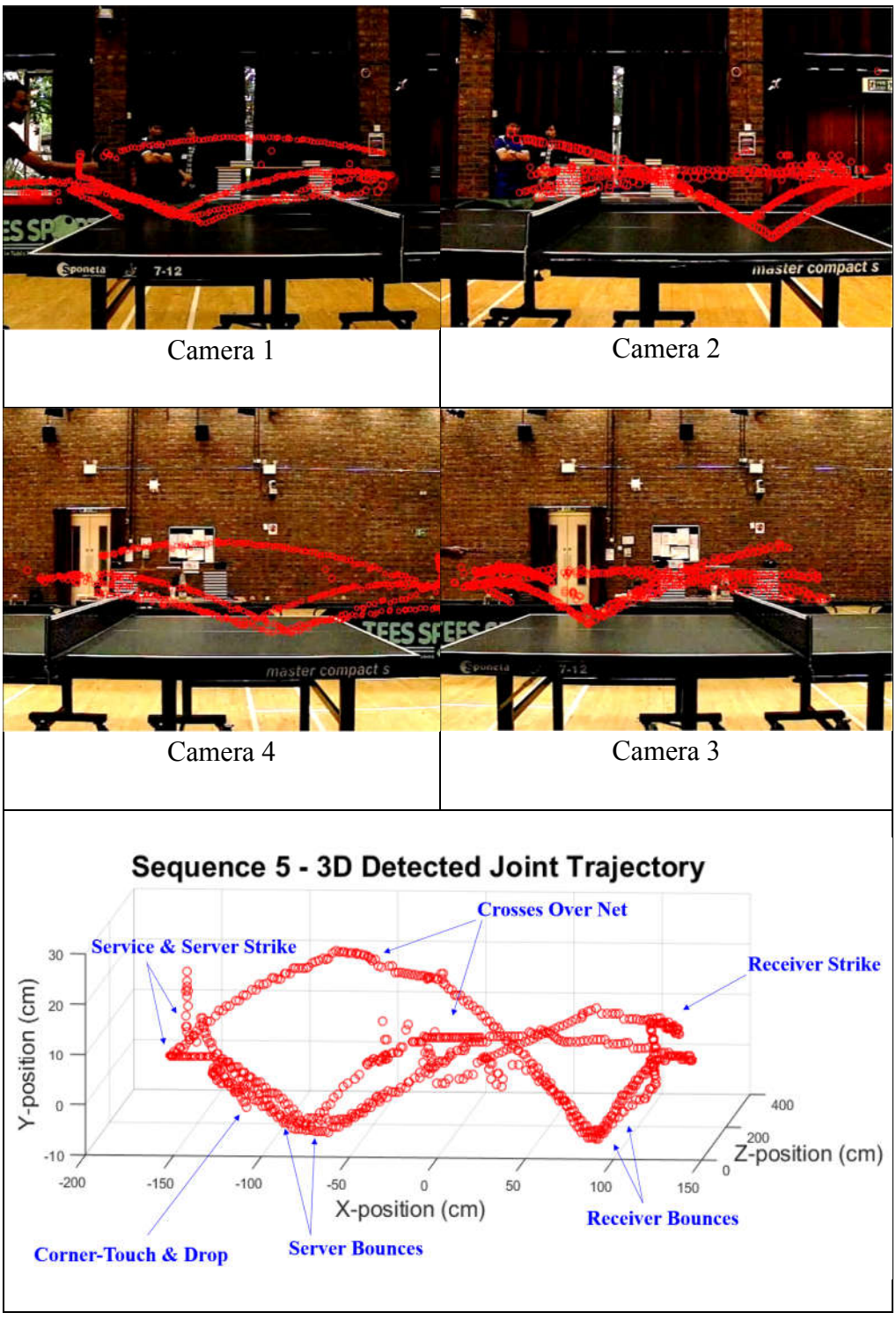


Figure 6.9: Sequence 5 - 3D Detected Joint Trajectory

```
INFO: ----------------------------------------
Agent container Main-Container@192.168.100.2 is ready.
----------------------------------------
```

Player 1 (Seq: 2)	Player 2
Serve →	→ Return
Return ←	→ Return
Return ←	
U: Under Table Player 1 is a Loser	

```
PC2 : NoBall received
PC2 : Informing failure to IVCA

SD : Frame : 5 : State : 4 : Description : Falling
SD : Frame : 20 : State : 5 : Description : ServerStrike
SD : Frame : 65 : State : 6 : Description : ServerBounce
SD : Frame : 119 : State : 7 : Description : CrossOverNet(StoR)
SD : Frame : 187 : State : 8 : Description : ReceiverBounce
SD : Frame : 277 : State : 9 : Description : ReceiverStrike
SD : Frame : 343 : State : 10 : Description : CrossOverNet(RtoS)
SD : Frame : 386 : State : 11 : Description : ServerBounce
SD : Frame : 456 : State : 12 : Description : ServerStrike
SD : Frame : 545 : State : 7 : Description : CrossOverNet(StoR)
SD : Frame : 606 : State : 8 : Description : ReceiverBounce
SD : Frame : 711 : State : 9 : Description : ReceiverStrike
SD : Frame : 769 : State : 10 : Description : CrossOverNet(RtoS)
SD : Frame : 798 : State : 11 : Description : ServerBounce
IVCA : For Area :1 The ball disappeared from both view                : Sleep
```

Figure 6.10: Sequence 5 - Results of multi-agent system

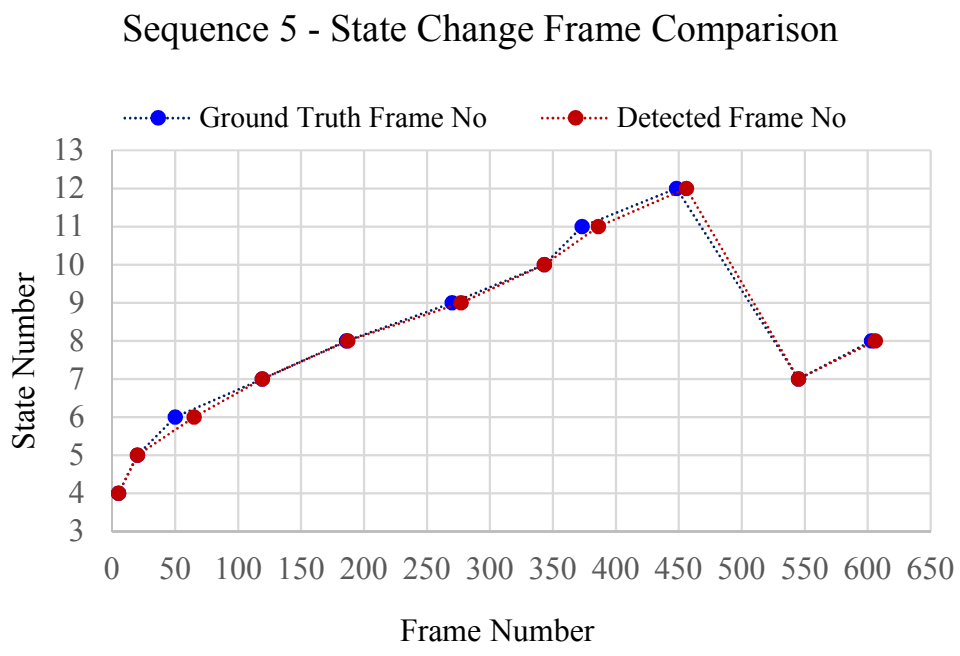


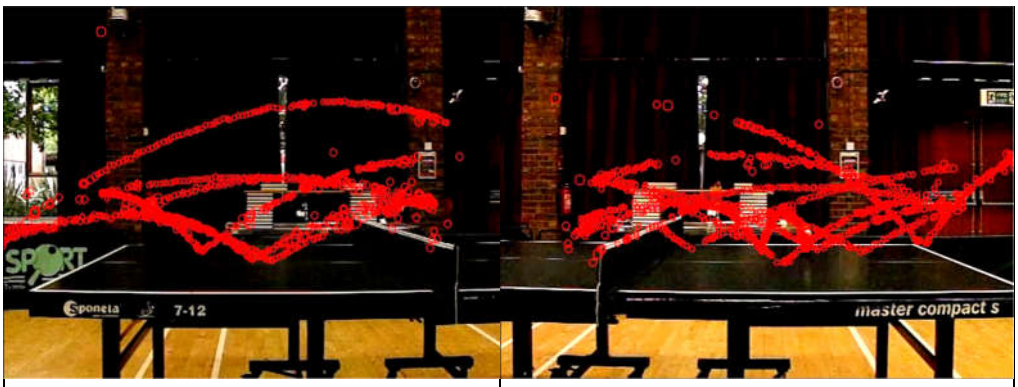
Figure 6.11: Sequence 5 - State Change Frame Comparison

6.5.3 Sequence 6: Results Discussion

Among the ten sequences, the sequence 6 is the longest sequence which is composed of 1800 frames. The system can identify every state correctly with the detailed 3D ball detection results with the average of 2 frames delay. The rally is ended by the ball crosses over the net and has multiple bounces on the table. To get the overall idea, the table 6.4 shows the state change frame comparison, figure 6.11 shows the joint trajectory history of BDA1 and BDA2. The figure 6.12 and 6.13 provide the results of a multi-agent system.

Table 6.4: Sequence 6 - State Change frame comparison

State	Ground Truth Frame No	Detected Frame No	Frame Delay	Average Frame Delay
6	35	35	0	2
7	105	105	0	
8	160	154	6	
9	180	170	10	
10	313	313	0	
11	369	370	1	
12	450	450	0	
7	529	529	0	
8	606	600	6	
9	640	640	0	
10	752	752	0	
11	810	810	0	
12	869	875	6	
7	952	952	0	
8	992	998	6	
9	1078	1078	0	
10	1213	1213	0	
11	1311	1313	2	
12	1483	1483	0	



Camera 1 | Camera 2

Camera 4 | Camera 3

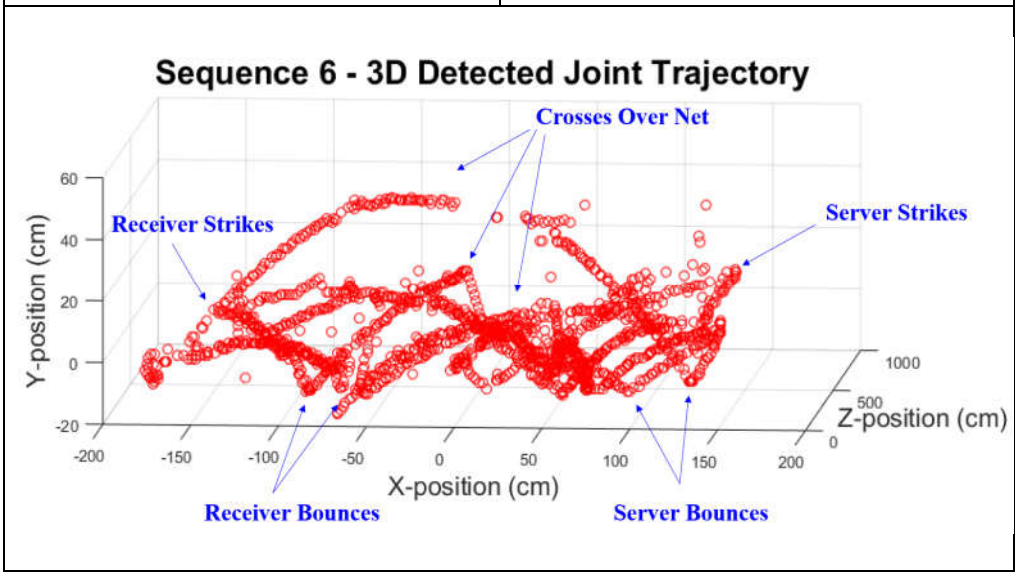


Figure 6.12: Sequence 6 - 3D Detected Joint Trajectory

```
INFO: ----------------------------------------
Agent container Main-Container@137.108.123.10 is ready.
----------------------------------------------
```

Player 1	Player 2 (Seq: 3)
Return	Serve
Return	Return
Return	Return
	Return
	M: Multiple Bounces Player 2 is a Loser

```
SD : ServerSide : 1
SD : Frame : 1 : State : 5 : ServerStrike : PC2, X: 148, Y: 9, Z: 120

PC1 : NoBall received
PC1 : Informing failure to IVCA
SD : Frame : 35 : State : 6 : ServerBounce : PC2, X: 92, Y: -9, Z: 88
SD : Frame : 105 : State : 7 : CrossNet(StoR) : PC2, X: 5, Y: 10, Z: 108
SD : Frame : 154 : State : 8 : ReceiverBounce : PC1, X: -52, Y: -8, Z: 113
SD : Frame : 170 : State : 9 : ReceiverStrike : PC1, X: -66, Y: -8, Z: 26
SD : Frame : 313 : State : 10 : CrossNet(RtoS) : PC2, X: -5, Y: 17, Z: 58
SD : Frame : 370 : State : 11 : ServerBounce : PC2, X: 62, Y: -9, Z: 75
SD : Frame : 450 : State : 12 : ServerStrike : PC2, X: 134, Y: 1, Z: 148
SD : Frame : 529 : State : 7 : CrossNet(StoR) : PC1, X: 0, Y: 26, Z: 83
SD : Frame : 600 : State : 8 : ReceiverBounce : PC1, X: -80, Y: -8, Z: 87
SD : Frame : 640 : State : 9 : ReceiverStrike : PC1, X: -112, Y: 7, Z: 122
SD : Frame : 752 : State : 10 : CrossNet(RtoS) : PC2, X: -5, Y: 17, Z: 111
SD : Frame : 810 : State : 11 : ServerBounce : PC2, X: 84, Y: -8, Z: 106
SD : Frame : 875 : State : 12 : ServerStrike : PC2, X: 127, Y: 13, Z: 88
SD : Frame : 952 : State : 7 : CrossNet(StoR) : PC2, X: 3, Y: 7, Z: 95
SD : Frame : 998 : State : 8 : ReceiverBounce : PC1, X: -59, Y: -9, Z: 207
SD : Frame : 1078 : State : 9 : ReceiverStrike : PC1, X: -143, Y: 1, Z: 53
SD : Frame : 1213 : State : 10 : CrossNet(RtoS) : PC1, X: -5, Y: 51, Z: 84
SD : Frame : 1313 : State : 11 : ServerBounce : PC2, X: 135, Y: -6, Z: 130
SD : Frame : 1483 : State : 12 : ServerStrike : PC2, X: 142, Y: 25, Z: 123
SD : Frame : 1773 : State : 12 : Faulty:MultipleBounces : PC2, X: 73, Y: -9, Z: 96
SD : Frame : 1778 : State : 12 : Faulty:MultipleBounces : PC2, X: 73, Y: -9, Z: 96
SD : Frame : 1783 : State : 12 : Faulty:MultipleBounces : PC2, X: 73, Y: -9, Z: 91
EOF received
```

Figure 6.13: Sequence 6 - Results of multi-agent system

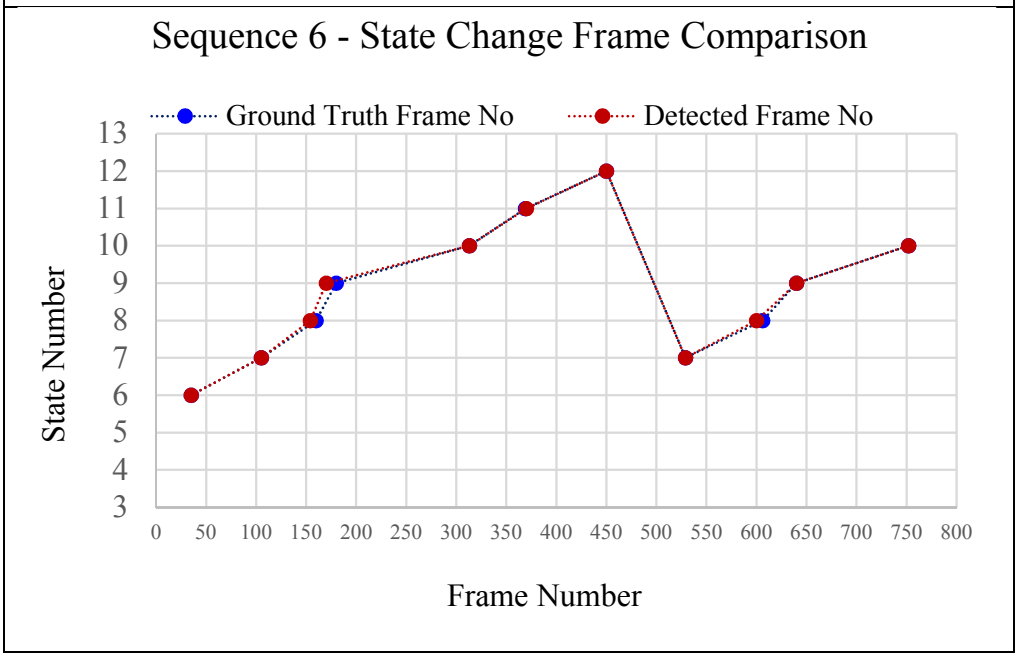


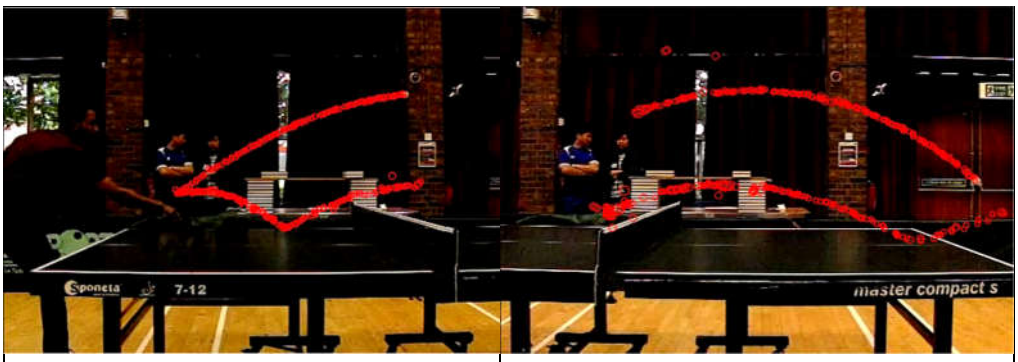
Figure 6.14: Sequence 6 - State Change Frame Comparison

6.5.4 Sequence 7: Results Discussion

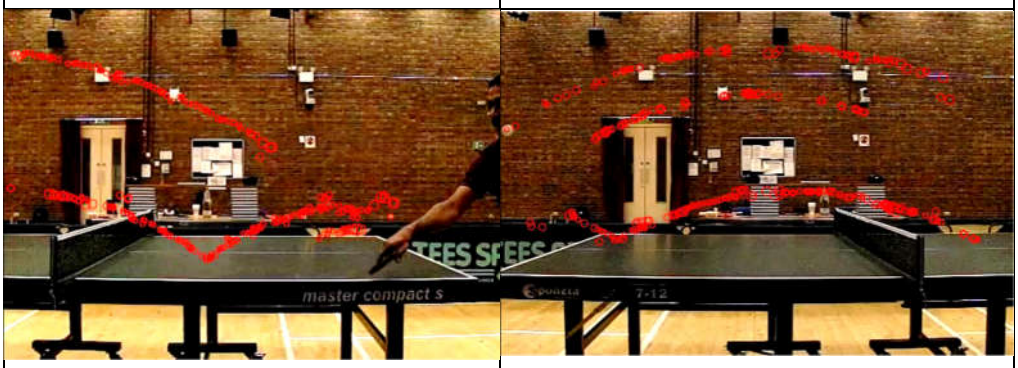
In this sequence, the ball is entering the camera view by state 5 (Beginning Server Strike) and the system is detected from state 6 (Server Bounce) to state 12 (Looping Server Strike). The rally is ended by the ball goes beyond the server end-line without bouncing in the opponent's court after being struck by the opponent. As can be seen in the figure 6.14, the whole trajectory is constructed based on the results of BDA1 (PC1) and BDA2 (PC2). The figure 6.15 provide the results of a multi-agent system for sequence 4 in which the ball is travelling from state 5 to state 10. The table 6.5 and the figure 6.16 present the comparison of state change frame. The system can identify every state correctly with the detailed 3D ball detection results with the average of 3 frames delay.

Table 6.5: Sequence 7 - State Change frame comparison

State	Ground Truth Frame No	Detected Frame No	Frame Delay	Average Frame Delay
5	1	1	0	3
6	19	20	1	
7	110	108	2	
8	153	153	0	
9	180	168	12	
10	297	297	0	



Camera 1 | Camera 2



Camera 4 | Camera 3

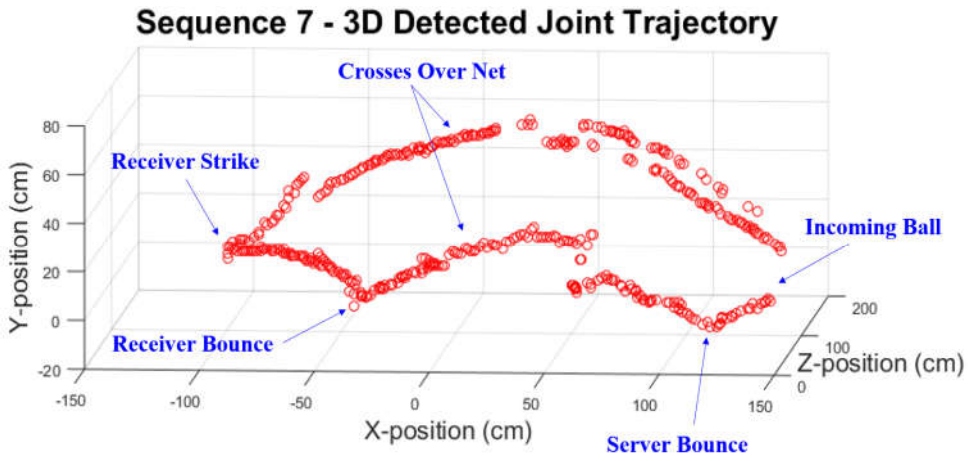


Figure 6.15: Sequence 7 - 3D Detected Joint Trajectory

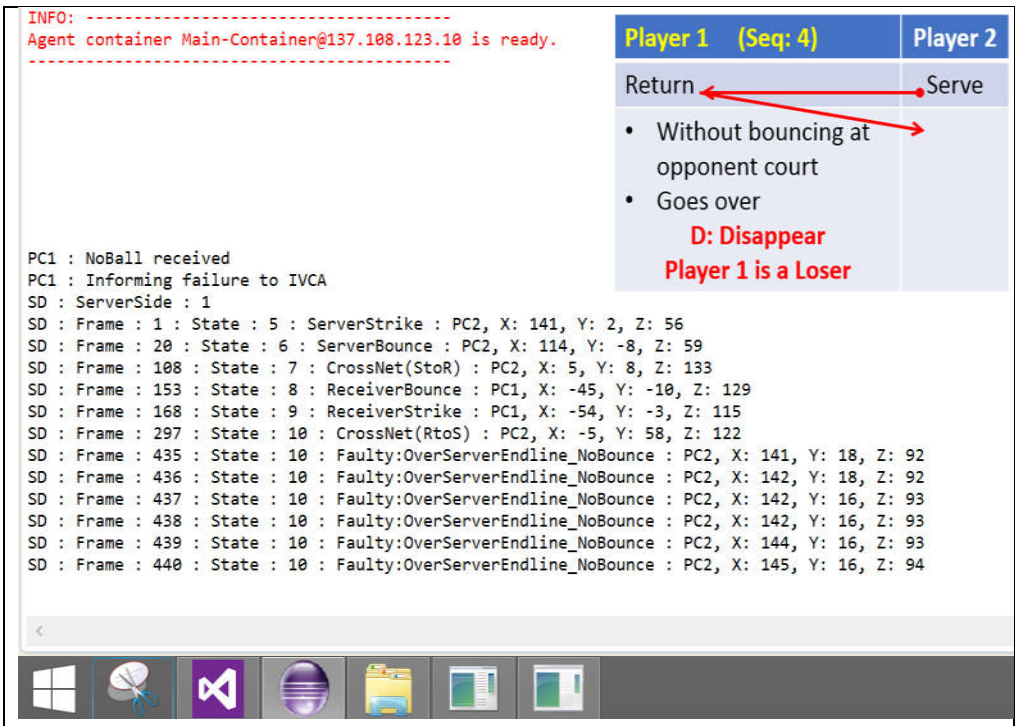


Figure 6.16: Sequence 7 - Results of multi-agent system

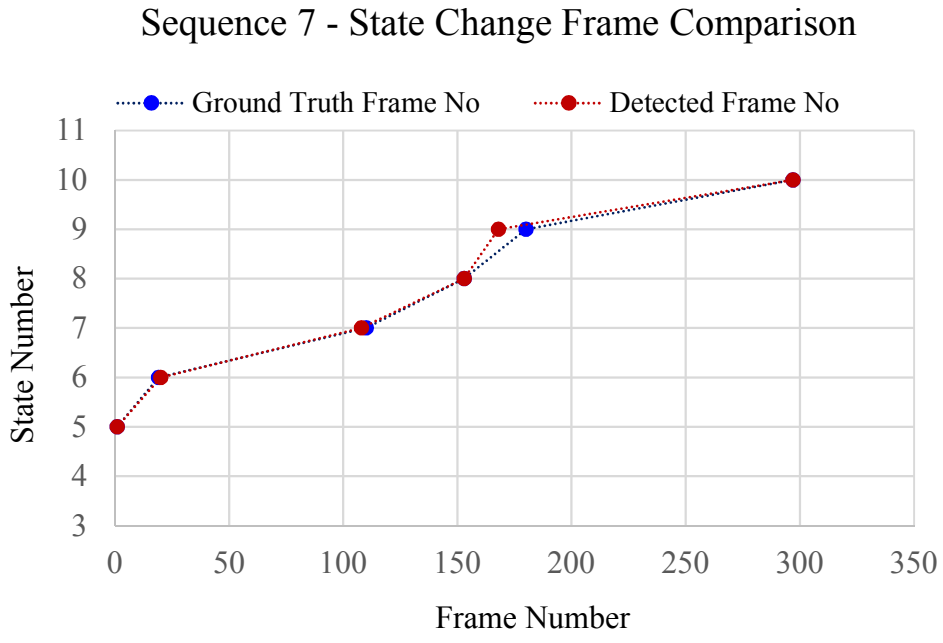


Figure 6.17: Sequence 7 - State Change Frame Comparison

6.5.5 Sequence 8: Results Discussion

In this sequence, the ball is entering the camera view by state 4 (Falling) and the system is detected from state 5 (Beginning Server Strike) until state 8 (Receiver Bounce). The rally is ended by the receiver misses the ball. The whole trajectory is constructed based on the results of BDA1 (PC1) and BDA2 (PC2). The system can identify every state correctly with the detailed 3D ball detection results with the average of 2 frames delay. To get the overall idea, the table 6.6 shows the state change frame comparison, figure 6.17 shows the joint trajectory history of BDA1 and BDA2, and figure 6.18 and 6.19 show the result of multi-agent system.

Table 6.6: Sequence 8 - State Change frame comparison

State	Ground Truth Frame No	Detected Frame No	Frame Delay	Average Frame Delay
5	1	1	0	2
6	10	15	5	
7	93	93	0	
8	166	170	4	
10	238	238	0	

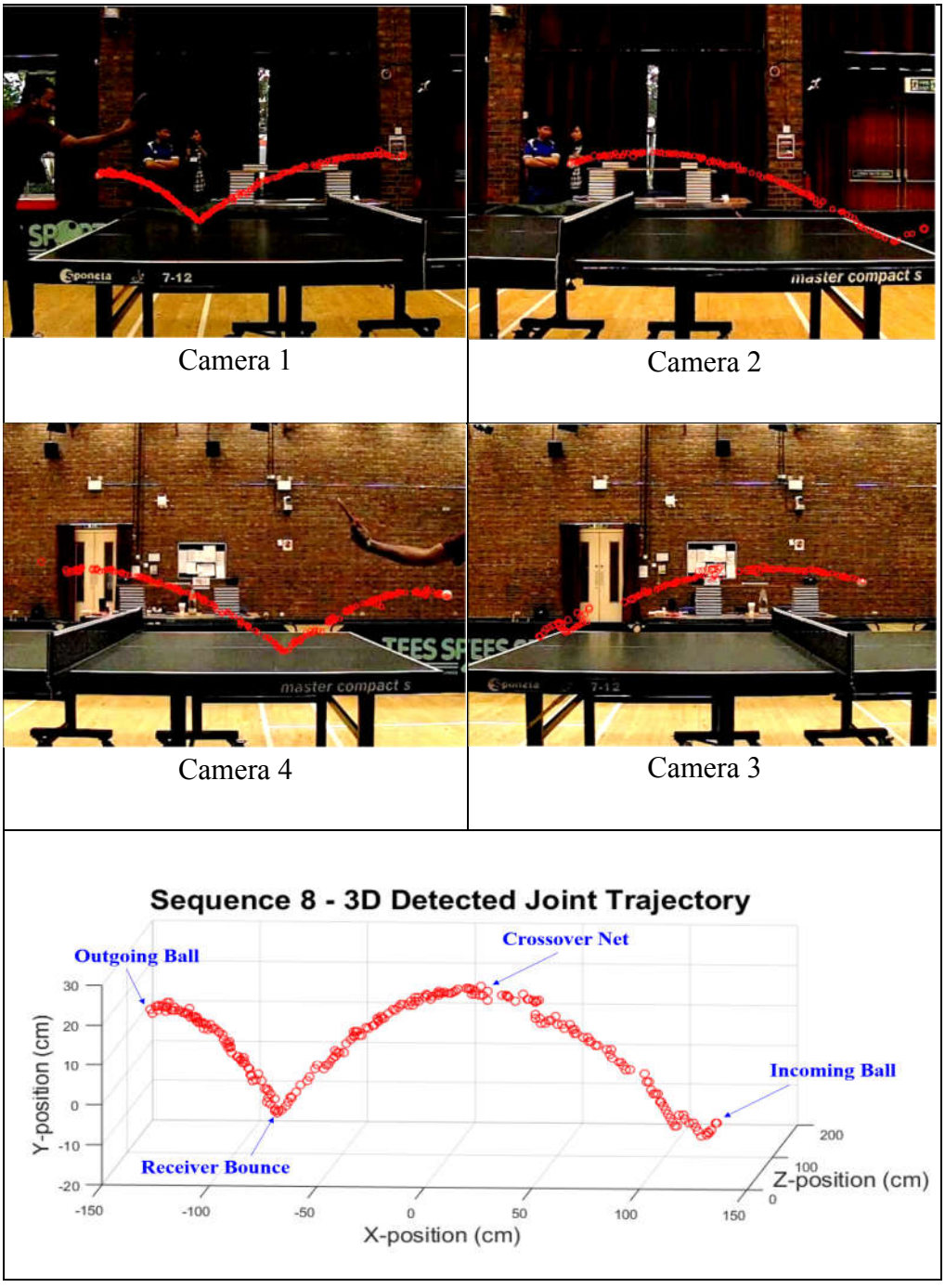


Figure 6.18: Sequence 8 - 3D Detected Joint Trajectory

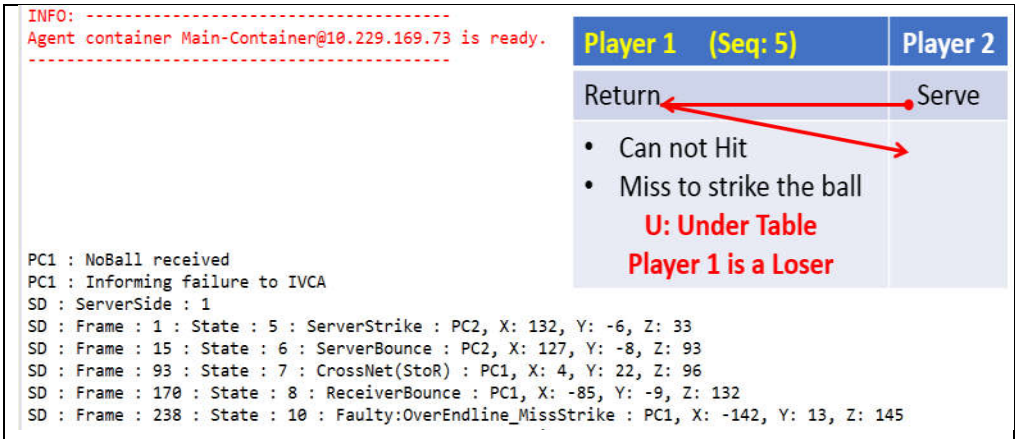


Figure 6.19: Sequence 8 - Results of multi-agent system

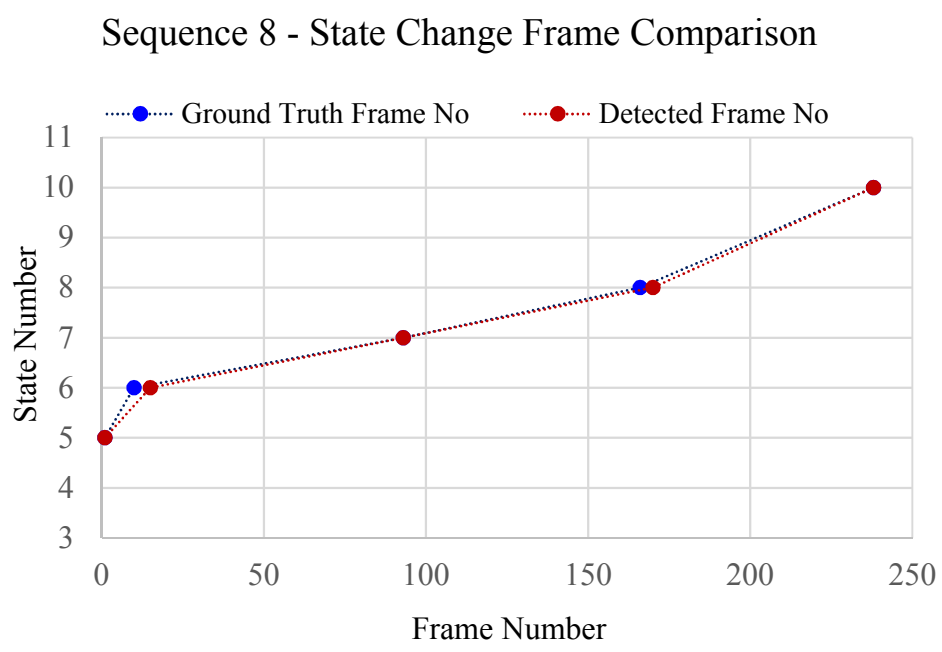


Figure 6.20: Sequence 8 - State Change Frame Comparison

6.5.6 Sequence 9: Results Discussion

In this sequence, the BDA1 can partially see the service and the ball progresses from state 1 (On Palm) to 10 (Crossover Net). The rally is ended by the ball touched the corner of the table and dropping under the table. As can be seen in figure 6.20 below, the system can detect the corner-touch and can identify every state correctly with the detailed 3D ball detection results with the average of 2 frames delay. To get the overall idea, the table 6.7 shows the state change frame comparison, figure 6.21 shows the joint trajectory history of BDA1 and BDA2, and figure 6.22 shows the result of multi-agent system.

Table 6.7: Sequence 9 - State Change frame comparison

State	Ground Truth Frame No	Detected Frame No	Frame Delay	Average Frame Delay
1	1	1	0	2
2	22	22	0	
3	40	43	3	
4	41	44	3	
5	102	105	3	
6	130	135	5	
7	188	188	0	
8	257	260	3	
9	319	320	1	
10	407	407	0	
10	510	510	0	

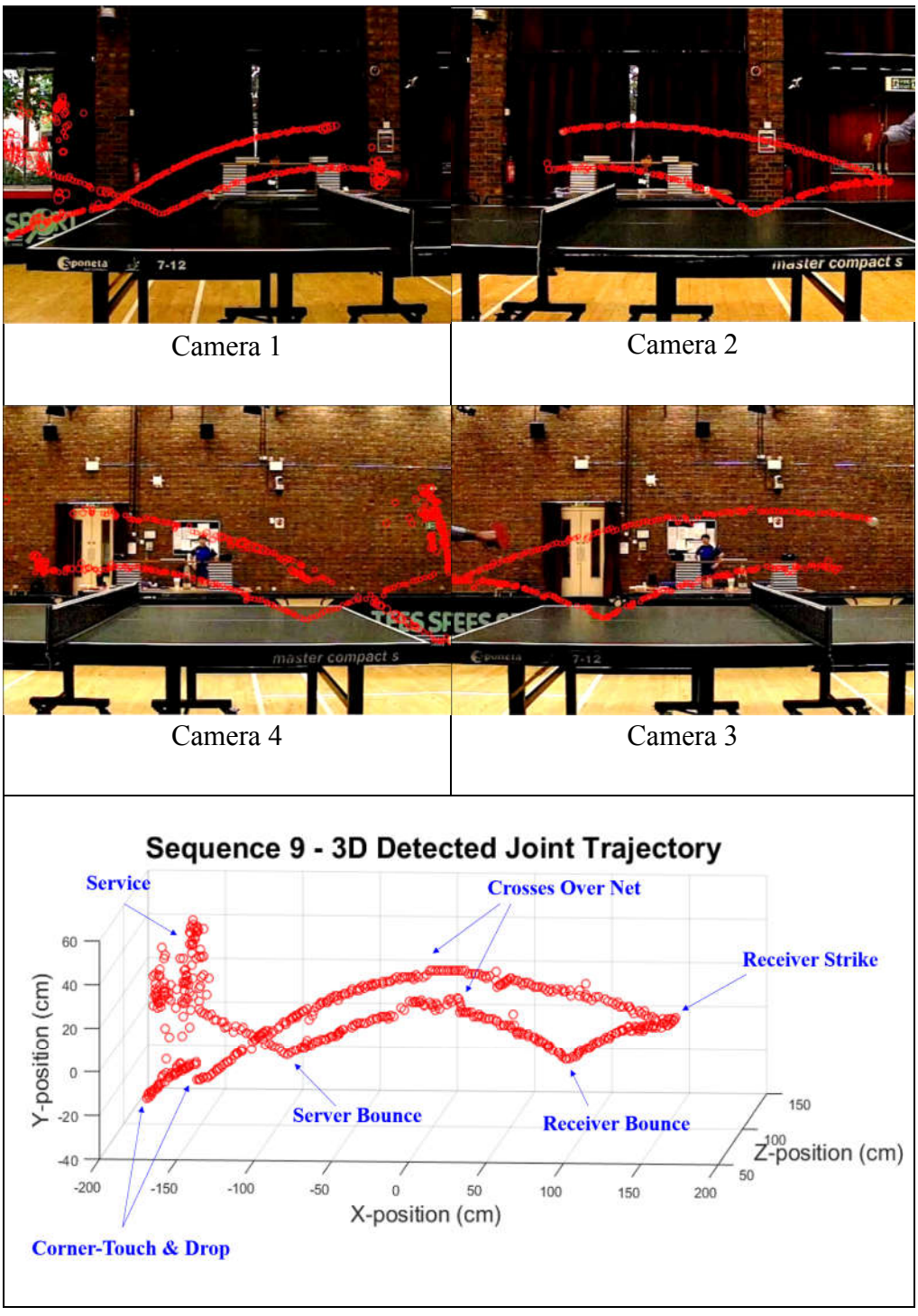


Figure 6.21: Sequence 9 - 3D Detected Joint Trajectory

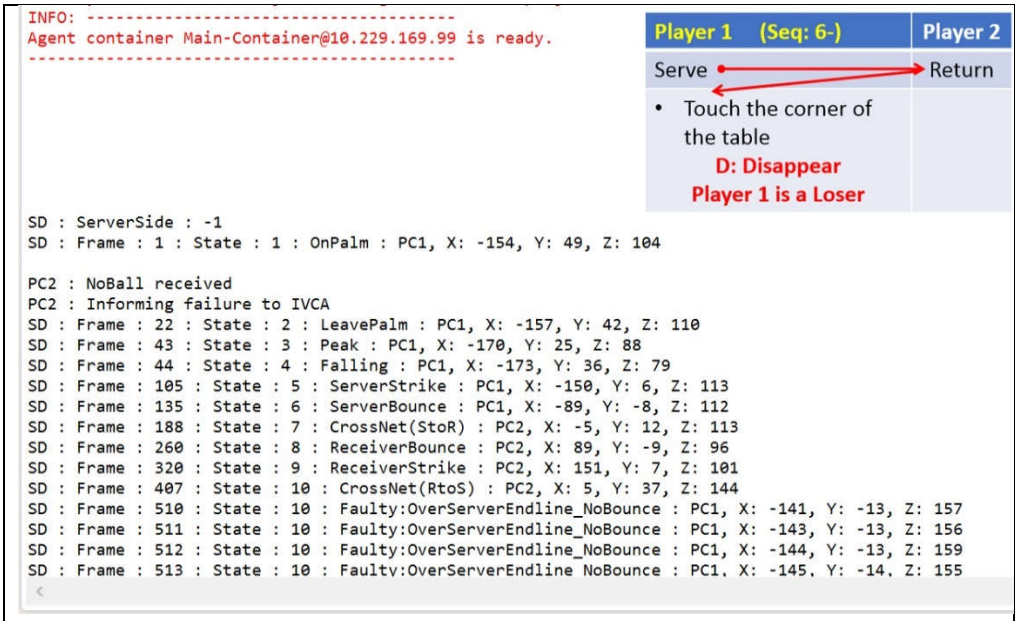


Figure 6.22: Sequence 9 - Results of multi-agent system

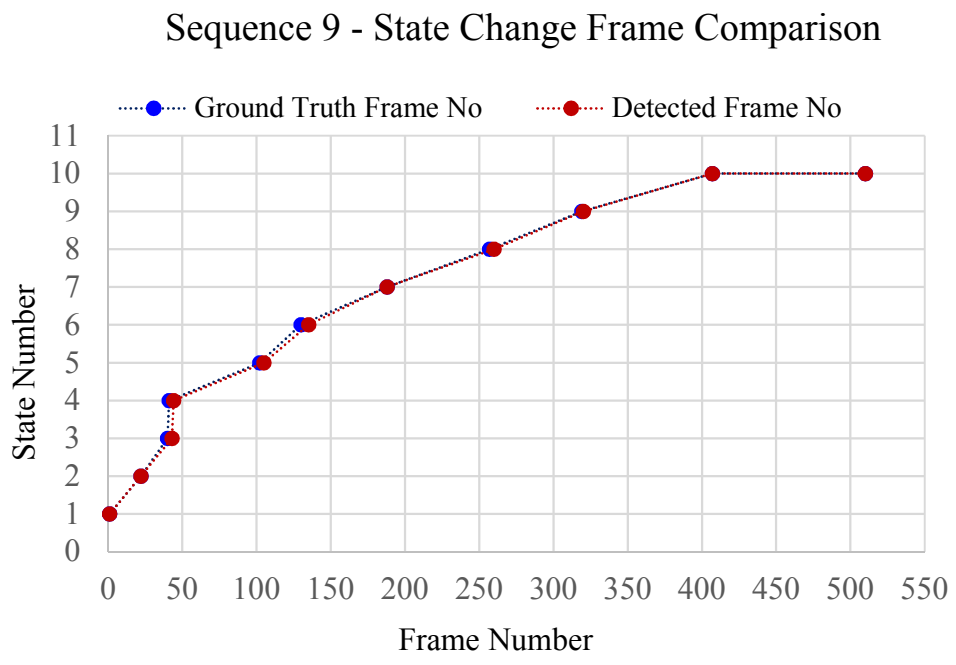


Figure 6.23: Sequence 9 - State Change Frame Comparison

6.5.7 Sequence 10: Results Discussion

Among the ten sequence, this sequence is filmed with very bright and the most complicated background. In this sequence, the ball is entering the camera view by state 4 (Falling) and the system is detected from state 5 (Beginning Server Strike) to state 12 (Looping Server Strike). The rally is ended by the ball crosses over the net and has multiple bounces on the table. As can be seen in the figure 6.23, the whole trajectory is constructed based on the results of BDA1 (PC1) and BDA2 (PC2). The system can identify every state correctly with the detailed 3D ball detection results with the average of 4 frames delay. To get the overall idea, the table 6.8 shows the state change frame comparison, figure 6.23 shows the joint trajectory history of BDA1 and BDA2, and the figures 6.24 and 6.25 shows the result of multi-agent system.

Table 6.8: Sequence 10 - State Change frame comparison

State	Ground Truth Frame No	Detected Frame No	Frame Delay	Average Frame Delay
5	1	1	0	4
6	43	45	2	
7	88	88	0	
8	171	173	2	
9	234	236	2	
10	336	336	0	
11	427	431	4	
12	510	528	18	
7	590	591	1	
8	673	674	1	
9	820	820	0	
10	911	911	0	
11	1000	986	14	

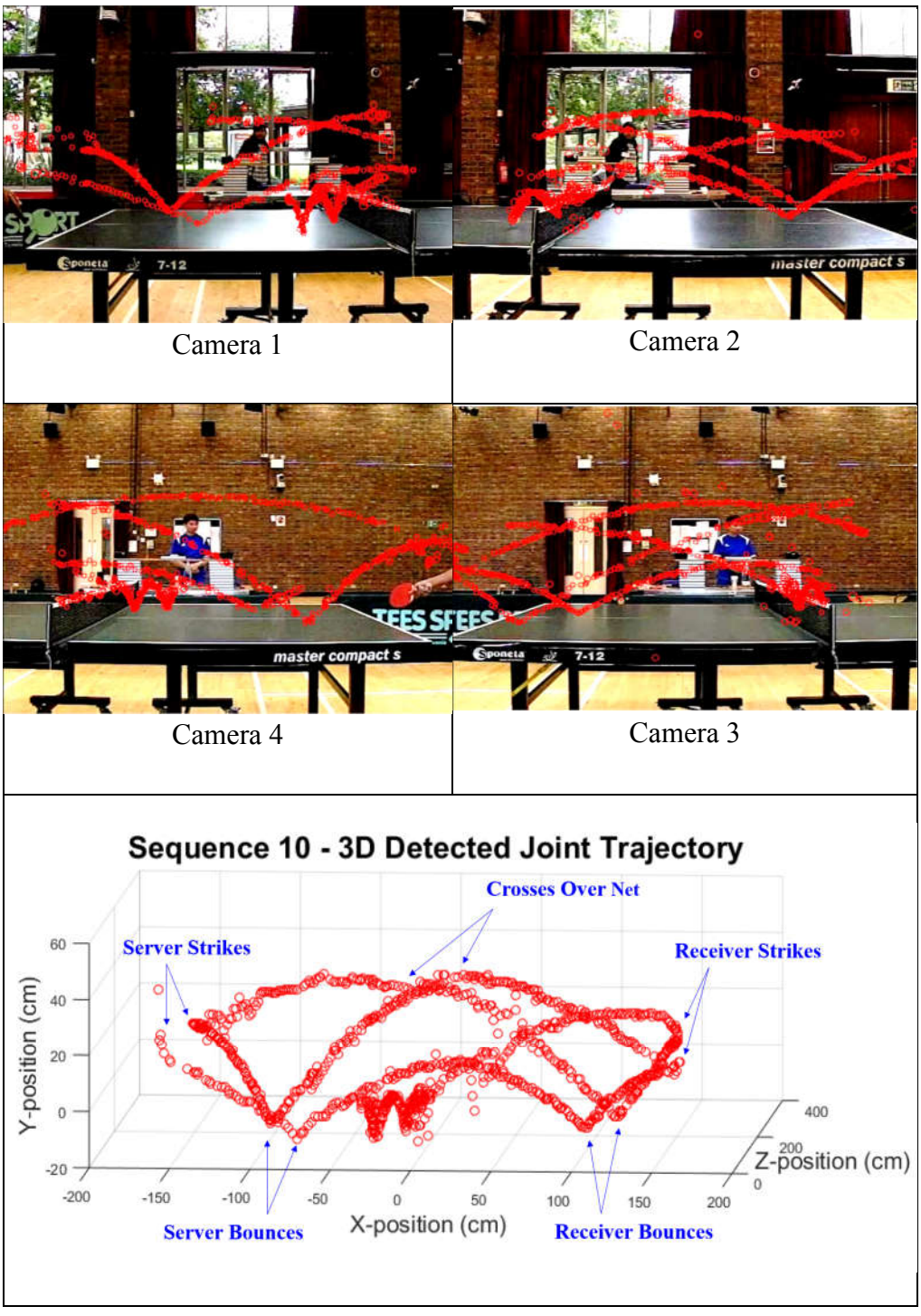


Figure 6.24: Sequence 10 - 3D Detected Joint Trajectory

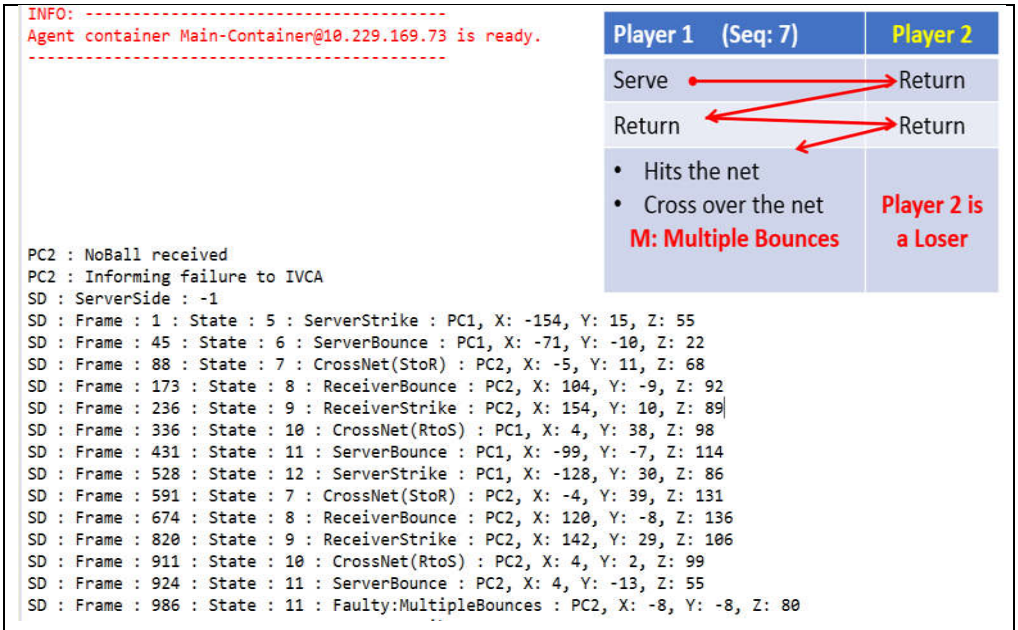


Figure 6.25: Sequence 10 - Results of multi-agent system

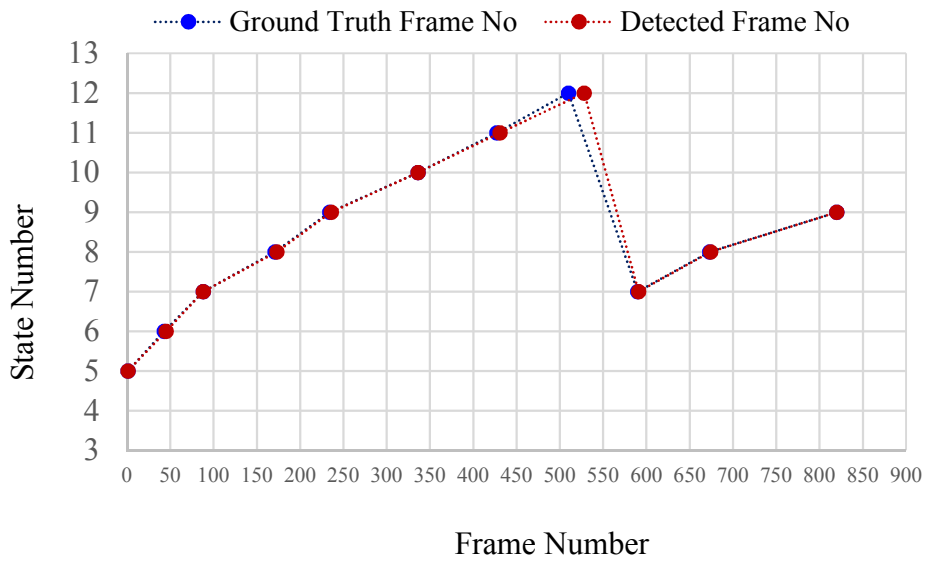


Figure 6.26: Sequence 10 - State Change Frame Comparison

6.6 Result Comparison

The system has been tested against the judgements of a human umpire. Both the accuracy and response rate have been considered. As shown in result discussion of each sequence, the system can identify different states of situations where a point was awarded, such as whether the ball was over the server end-line, bounced on the server side of the table or crossed over the net. Moreover, the system has an ability to declare when a fault occurs such as a fault due to double bounce, a fault due to a return not bouncing on the right side of the table, or a fault due to the ball drop under table and hitting the floor etc. Based on seven different sequences, the system can evaluate the entire rally through recorded video sequences of live play with 100% state detection rate with the average of 3 frames delay.

Table 6.9: System Average Frame Delay

Sequence Number	Number of Delay Frame in State Change	Average Frame Delay	State Detection Rate
4	2	3	100%
5	4		
6	2		
7	3		
8	2		
9	2		
10	4		

6.7 Summary

This chapter began with the main difficulties of tracking the ball in a match scene and the requirement of the incorporation of MAS. Subsequently, the multi-agent state machine was presented to critically analyse table tennis rallies and develop decision making techniques for providing a judgement. The proposed system was applied to a variety of complex table tennis rallies and some successful results were obtained. The system is able to track the ball, identify the state and can provide the decision about whether the ball goes under the playing surface or bounces multiple times on the table. As this is a pilot study, the focus is on the development of the techniques, rather than building a complete system. Therefore, some of the details of the table tennis rules will not be considered. As table tennis rules are revised from time to time, they may be changed in the future. The system is therefore implemented as a rule-based system to allow the efficient update of the rules.

Chapter 7

Future Work

There are a number of potential opportunities to extend the multi-view umpiring framework presented in this thesis, as well as exploring other domains for its applicability. Some of these prospective research avenues will now be reviewed.

1. The presented work had the clear overarching objective of achieving umpiring system without incurring high computational cost. Table tennis is the selected game to test the idea of this research. This thesis has constructed a complete ball's trajectory from multiple cameras, determined the legality of it, evaluated it according to the rules and decided autonomously by identifying the different states of rally. The system can identify a number of fault conditions, e.g. fault due to multiple bounces, fault due to not bouncing on the right side of the table, fault due to the ball dropping to the floor etc. Because of a limited time-frame of PhD research, this study was focused on developing the algorithms and techniques, rather than building a complete commercial product. Therefore, some states and situations were not cover in this research such as doubles player's rules and scoring the match. It would be insightful to investigate more detail states, more complex scenarios and continue to develop up to a complete umpiring system till

points are awarded. The addition of a set of agents that covers all the rules of the table tennis is the obvious first extension of this research.

2. Moreover, it is crucial to achieve a result as real-time to umpire a match. Another important future work is to develop a mechanism to reduce the computation. One possible way is to implement a new strategy which dynamically adjust the detection frame rate depends on the speed of the ball. When the ball travels slow, the rate of change of its velocity and acceleration is not much different and it is not necessary to detect the ball at every single available frame. By skipping some frames, it could reduce the processing time.
3. When the environment is complex and the needs of capturing the same object with different background, the contributed 3D derivation exploiting the proven FTF camera configuration can be used for identifying the real-world location of object. Moreover, the presented technique of multi-view auto correction and continuous tracking across multiple cameras observed by multi-agent approach can be extended to multiple moving objects tracking instead of one.
4. However, the current limitation of FTF cameras configuration is it fails to work out the 3D position of the ball when the ball is near the plane where the principal points of the opposite facing cameras joined. When the ball is in that region, the developed system has been extrapolated the ball positions by using the result from previous frames. A better solution is capturing the scenes with a larger OR between the SBS and to develop a strategy to

intelligently choose between using the FTF and SBS cameras for working out the 3D position.

5. The ability to detect and evaluate the correspondences 3D position between the object and its reference points can also be applied in rule-based decision-making system. One promising area of investigation would be to extend this framework to 3D automatic sport video umpiring such as football, cricket, tennis, volleyball, baseball, basketball, snooker, golf and etc. More functions could be inspired from the leading innovator in sports technology, Hawk-Eye which have gained popularity in the camera-based ball detection system due to its ability in tracking the ball, virtual display of it, providing the broadcast enhancement and assists officials when awarding points that played sports fairer and more engaging with the audience.

Chapter 8

Conclusion

Object tracking and evaluating its 3D position are increasingly used for innovative computer vision research including an automatic sport umpiring. To umpire a match automatically, first essential component is to detect the 3D position of object and use it to compare with the other reference points or lines for assessment. Although numerous methods have progressed rapidly in the last decades, detecting and tracking is still a challenging task when the object of interest (ball) has small size, fast movement across complex background, sudden change of trajectory and illumination changes. Due to this, the most useful features of object such as colour, shape and size become distorted and causes the detecting and tracking task to be difficult. Although powerful kernel-based tracking and classifier-based detection methods such as Kalman**,** Extended KF, Mean Shift Tracker**,** Particle filter and Support Vector Machine are widely employed in this area, they either imply a significant computational cost or time. While demanding reliable detection results for umpiring, conducting the whole tracking and evaluation process are required to finish in real-time. This limits the system not to develop with high computational algorithms. Although the proprietary decision support systems such as Hawk-eye provides good detection results, its hardware setup is fixed, expensive and

the need of an aerial view which are not applicable for the proposed low-cost and portable umpiring system.

In this thesis, a new multi-view ball detecting, and tracking strategy has been presented which comprises a suite of innovative algorithms to improve the detection performance by features based detecting and segmenting the object from the background using adaptive CT in combination with motion detection method. A new inter-view correction technique has been developed to recover the detection failure between different views. As well as, a new multi-agent framework has also been developed to enhance the tracking performance, lower the computational time complexity, simplicity, extendibility and it has been critically evaluated using several table tennis match sequences. In this way, this research makes three original contributions to the object detection, tracking and umpiring domain, which are summarised as follows:

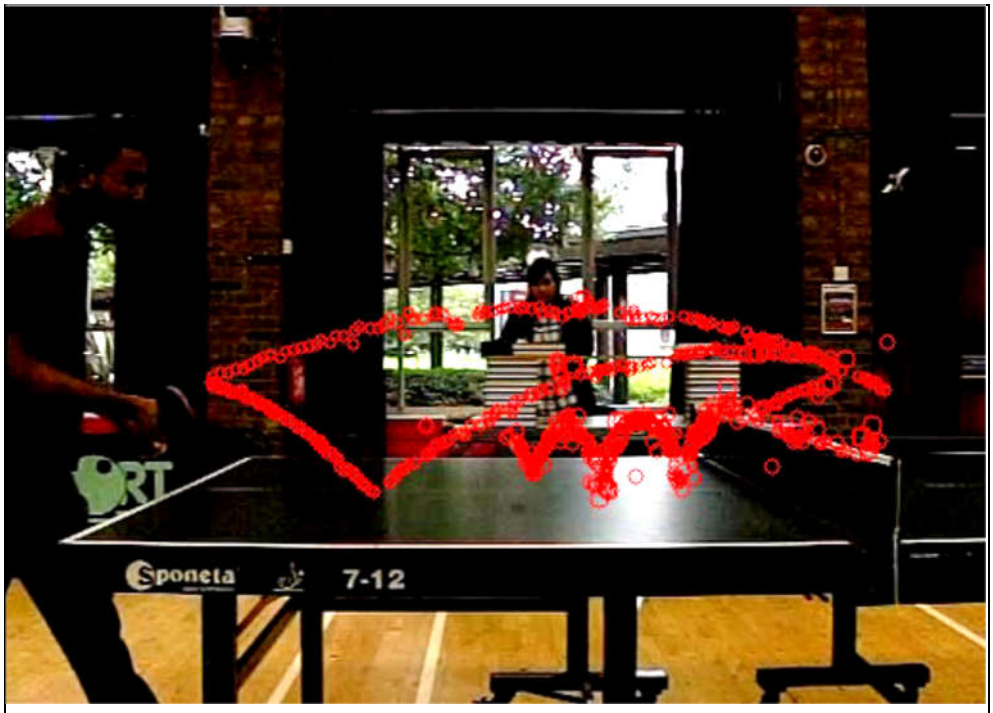
- The first significant contribution is a reliable ball detection strategy that accurately detects the location of ball in low resolution sequences. The ACTMD technique for object segmentation which has a capability to flexibly segment the ball from the background was developed for recomputing the threshold level and automatically updating between two effective visual and motion detection techniques based on the current results. Moreover, a new IVSC mechanism, which uses a positional information from another view to estimate the location of the object in current view, is developed to recover the detection failure. Rigorous experiments confirm that the IVSC improved the detection rate of individual views such as left view (83% detection rate) and right view (85% detection

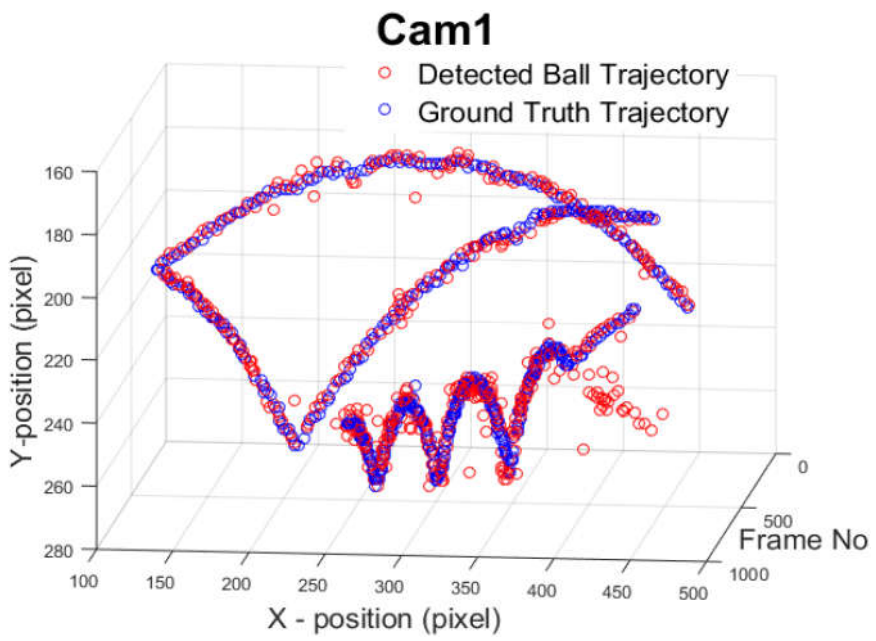
rate) to 97% (combined view detection rate) as discussed detail in Section 4.3.

- The second significant contribution is a novel framework for ball tracking using on a multi-view system which reduce occlusion, improve detection and continuous tracking the object among capturing devices. A new way of FTF camera configuration allows the system to capture the object at different angles, in either a closer or wider view. The developed Error Model reduces the calibration and misalignment errors from 10 cm to less than 1 cm as presented detail in Section 5.3. The development of artificial intelligence tracking technique allows the system to simultaneously process the data in parallel, automatically turn on and turn off the unnecessary cameras (that the ball is out of its view), identifies the favourable results when conflict results are occurred and reconstructs the broken trajectory to be a complete one. It provides the user with the flexibility to trade between speed and tracking ability.
- The third significant contribution is a new state-machine based evaluation system for analysing table tennis rallies. The developed intelligent system can analyse the table tennis rallies, automatically evaluate its legality and identify the current state of rally. This means the developed multi-agent umpiring system is computationally more efficient, adaptive and flexible in changes at will than the conventional umpiring system. Finally, several table tennis match sequences have been tested for evaluation and experimental results have been conclusively confirm that the average detection rates are over 94% among sequences and 100% umpiring results.

In summarising, this new automatic system for real-time video umpiring makes a notable contribution to the detecting, tracking and evaluating performance of video sequences characterised by illumination, noises and multiple occlusion conditions. Most importantly, it offers a flexible, adaptable and scalable solution for tracking, assisting decision and providing a measurement evidence not only be executed in sport video sequences, but also extendable to the 3D computer vision domain in the future.

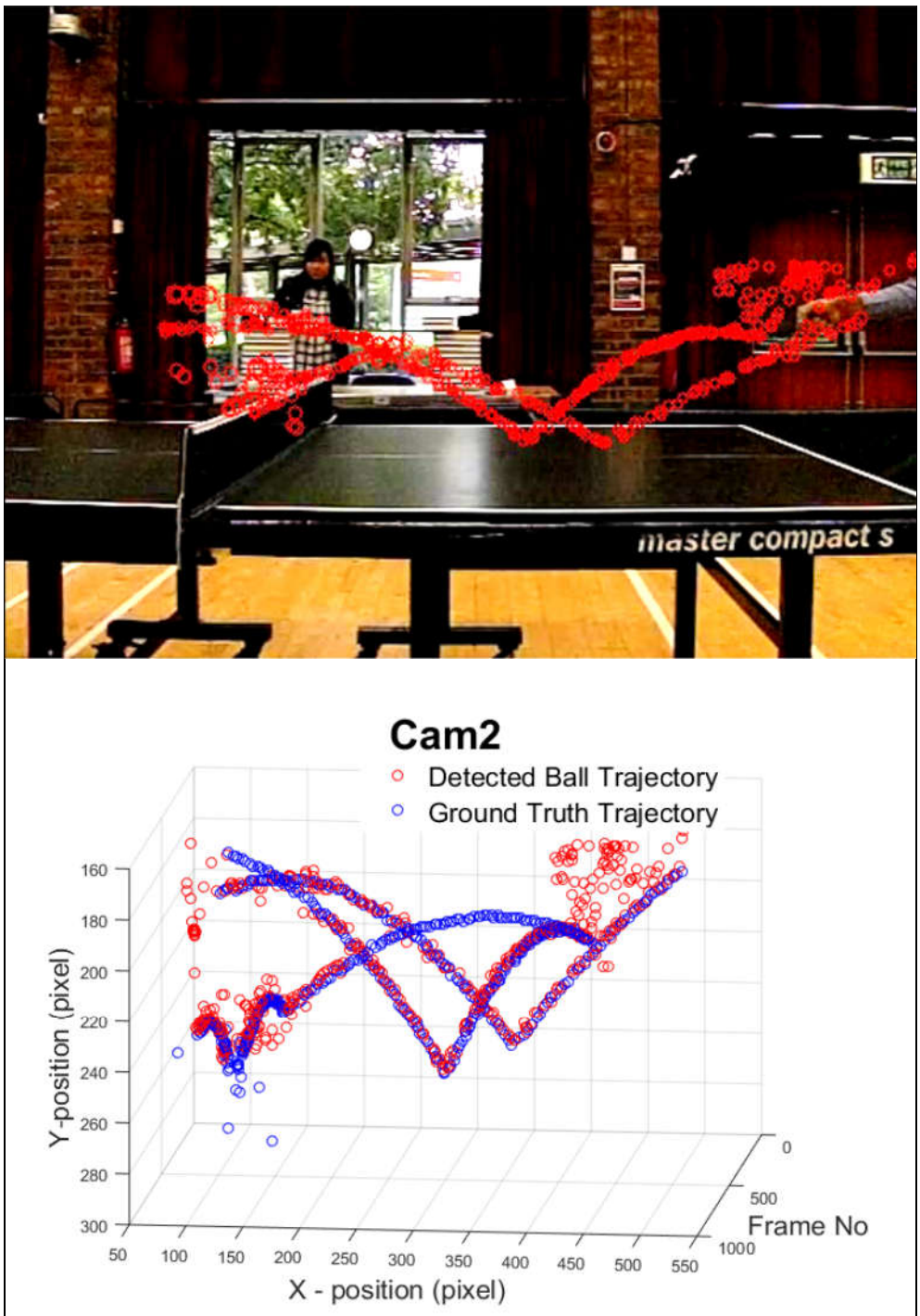
Appendix





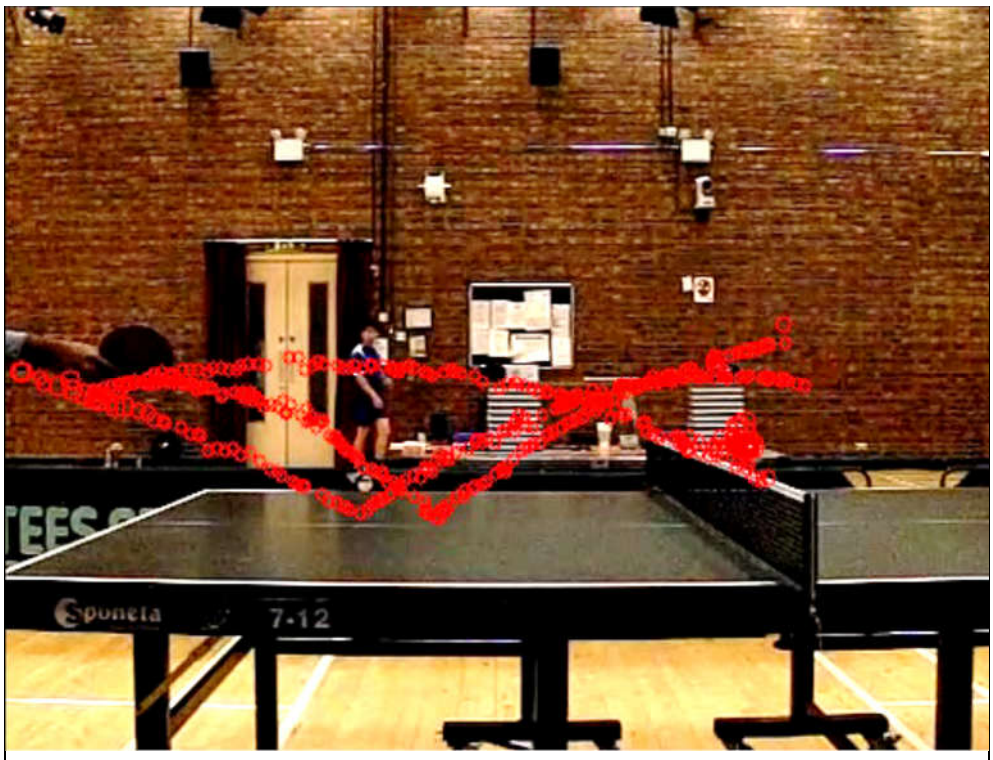
Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 1	876	92%	3.5 pixels	3.2 pixels	0.029 sec

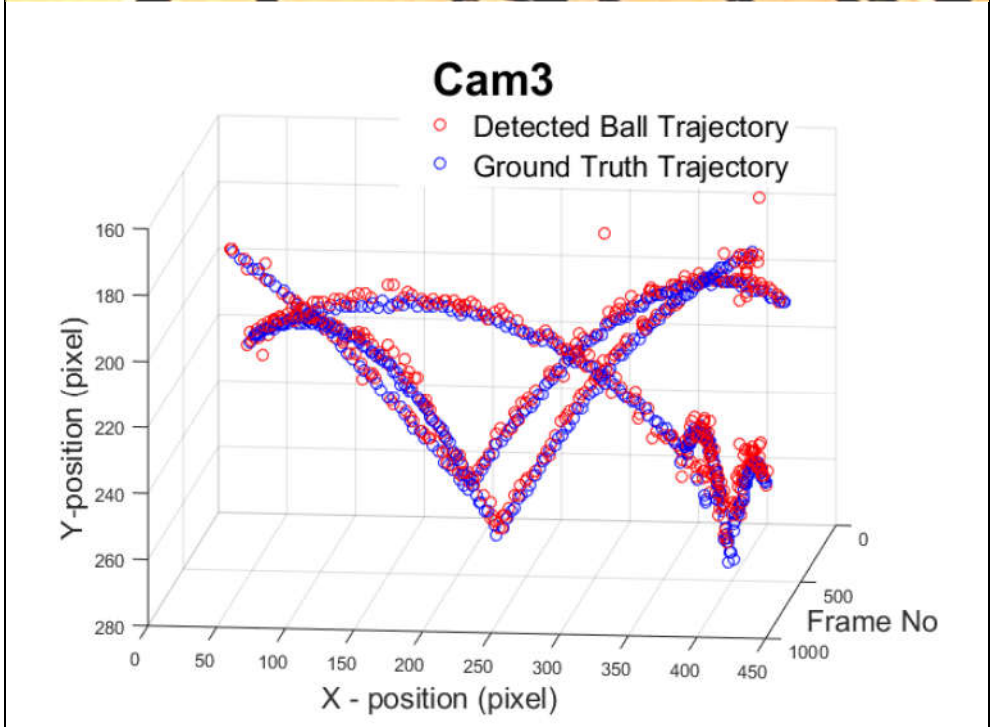
Figure A.1: Sequence 4 - Camera 1: 2D Trajectory comparison



Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 2	702	83%	3.5 pixels	3.6 pixels	0.024 sec

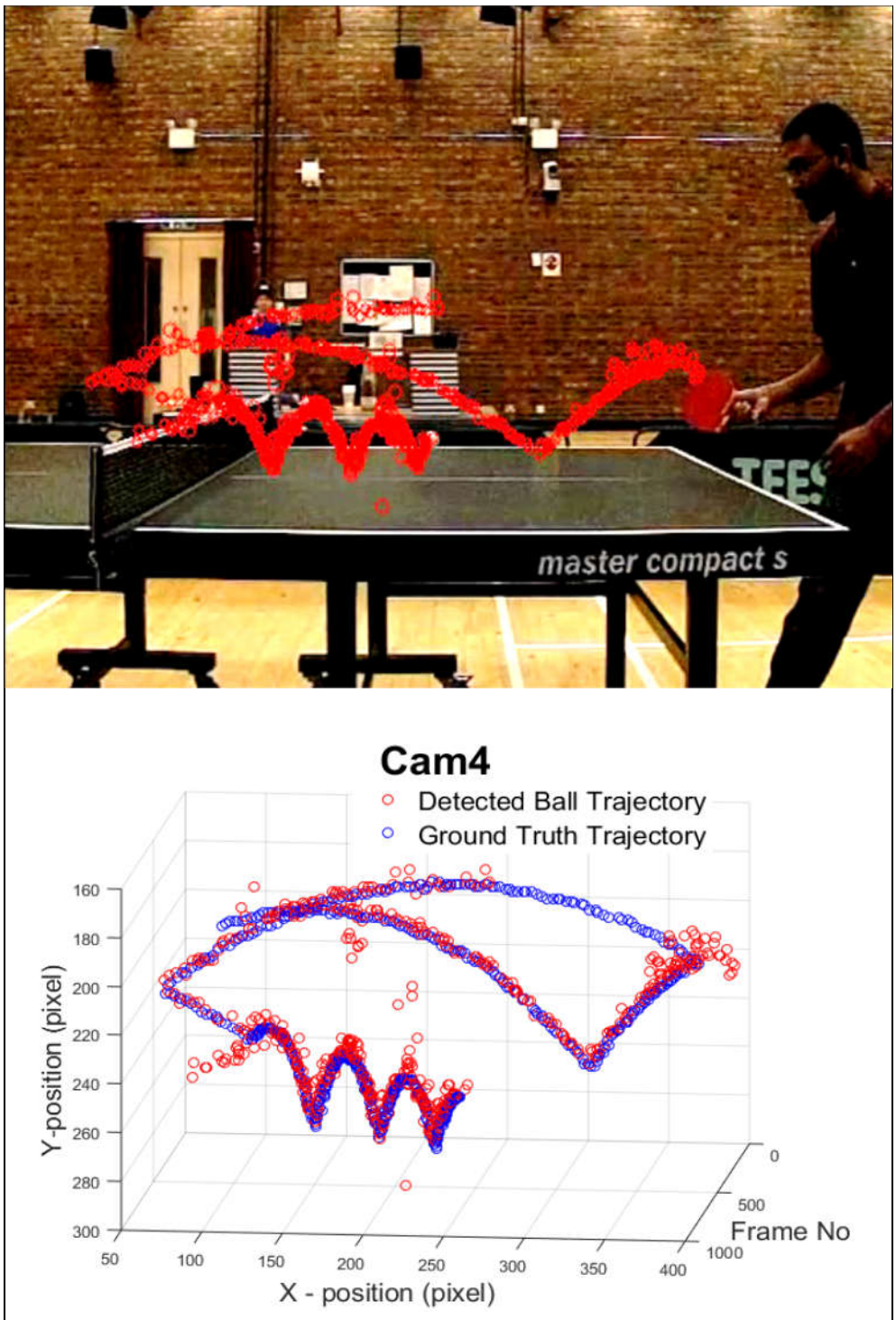
Figure A.2: Sequence 4 - Camera 2: 2D Trajectory comparison





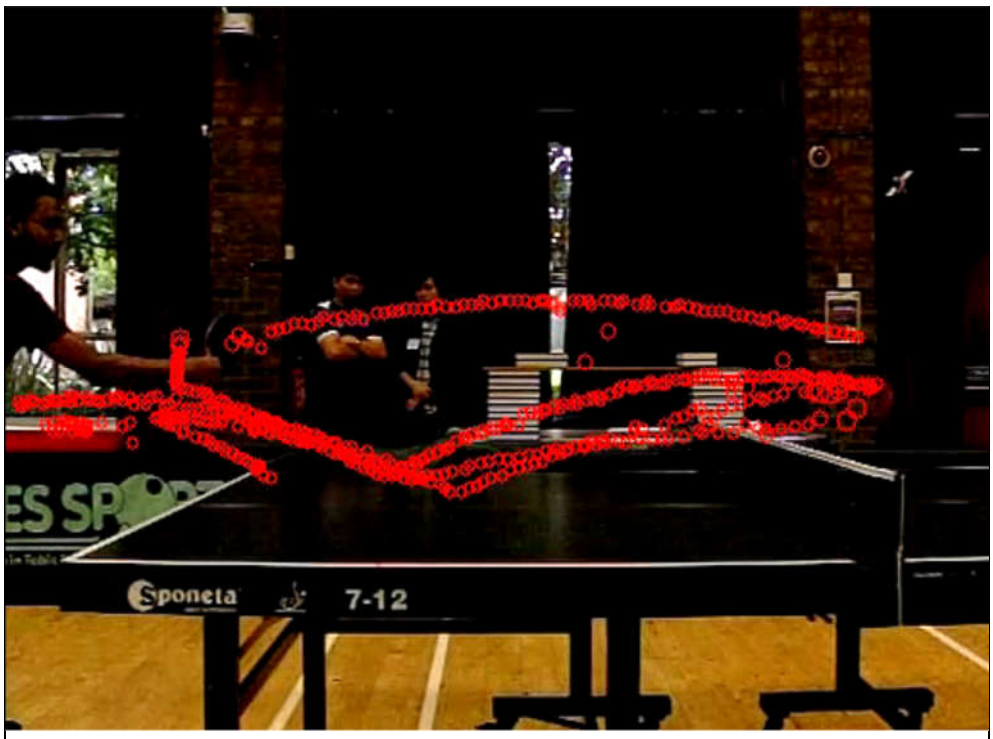
Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 3	702	95%	3.5 pixels	2.1 pixels	0.024 sec

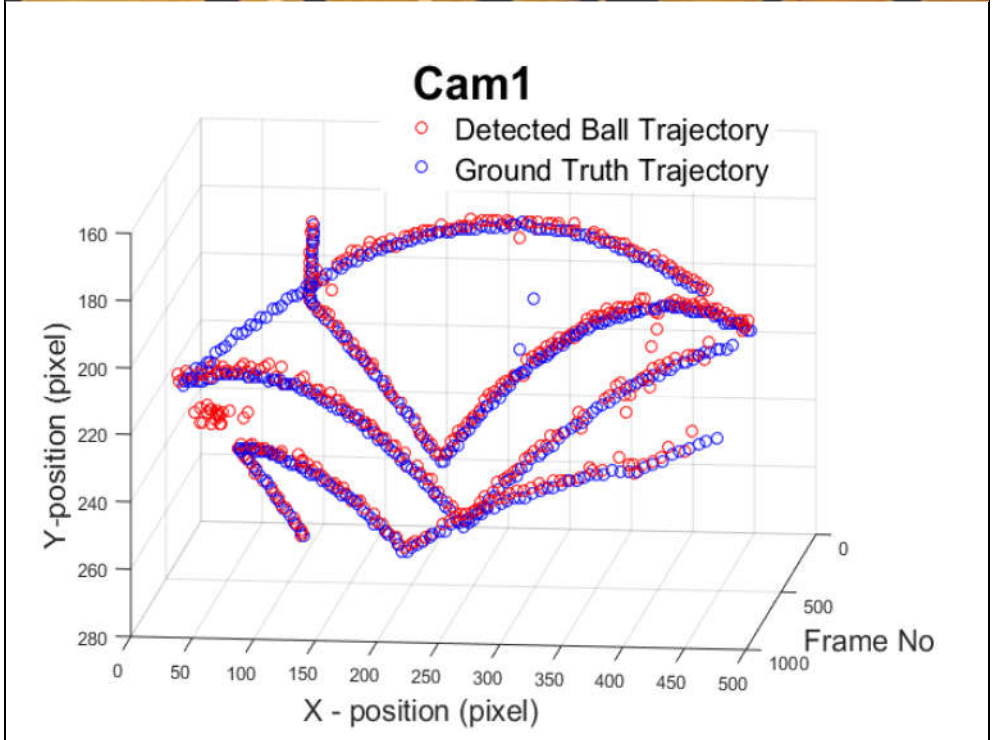
Figure A.3: Sequence 4 - Camera 3: 2D Trajectory comparison



Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 4	876	90%	3.5 pixels	3.9 pixels	0.029 sec

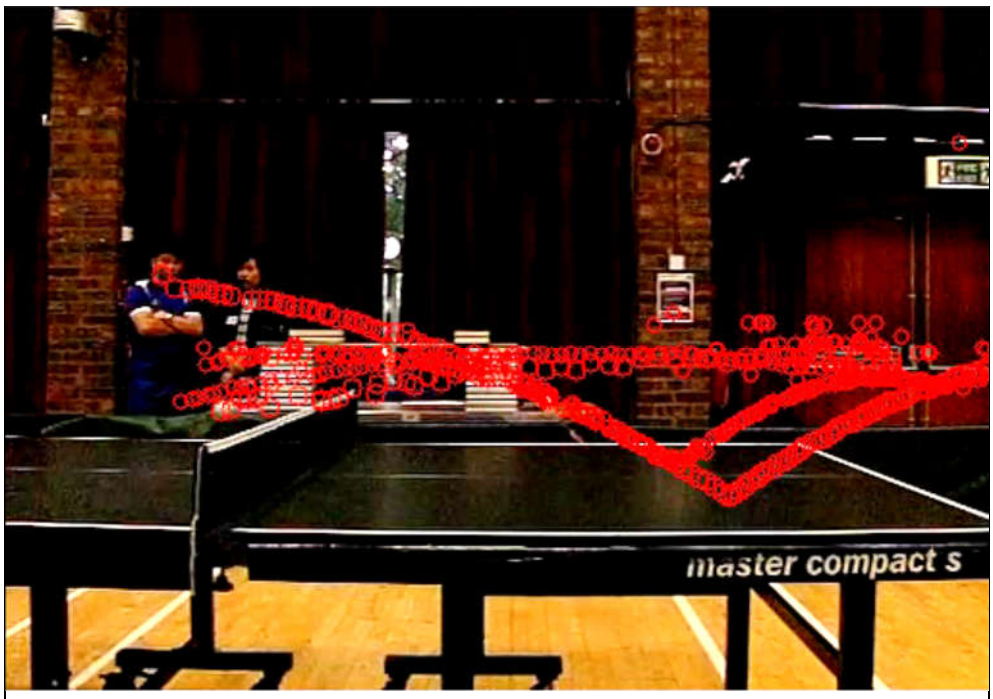
Figure A.4: Sequence 4 - Camera 4: 2D Trajectory comparison

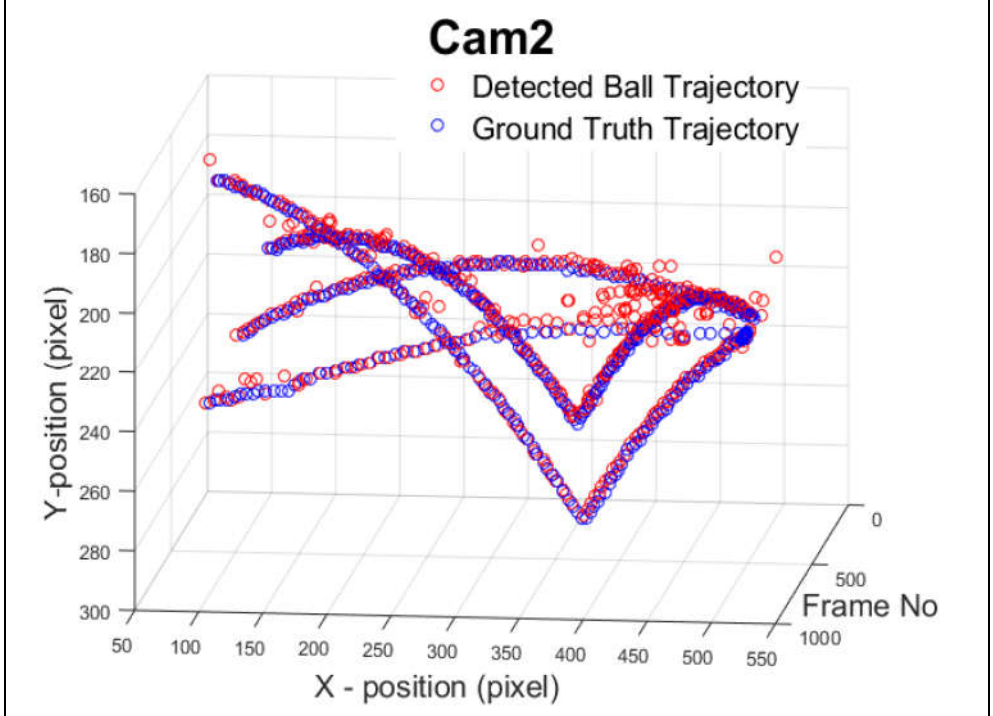




Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 1	860	96%	3.5 pixels	3.2 pixels	0.028 sec

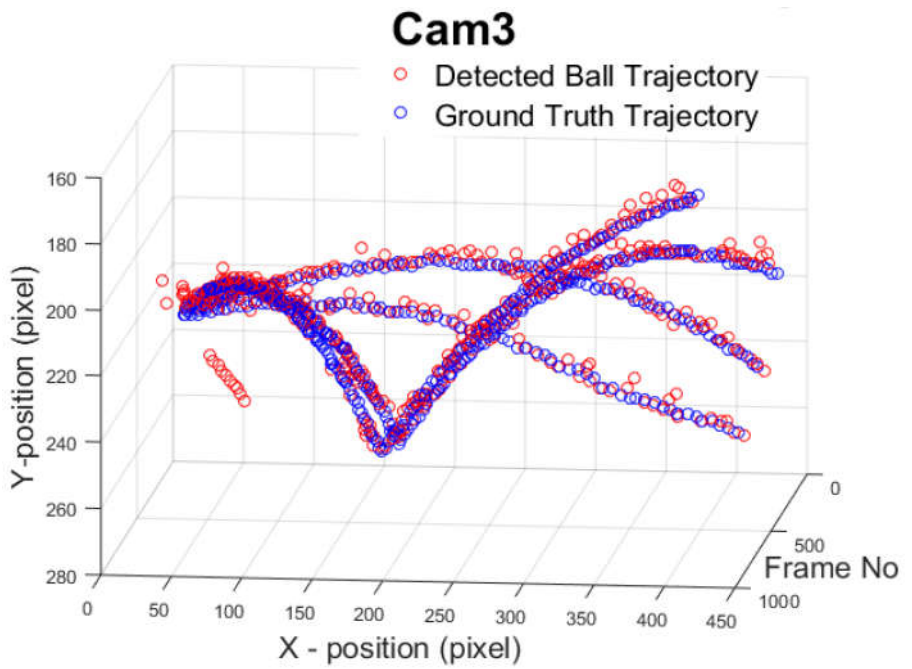
Figure A.5: Sequence 5 - Camera 1: 2D Trajectory comparison





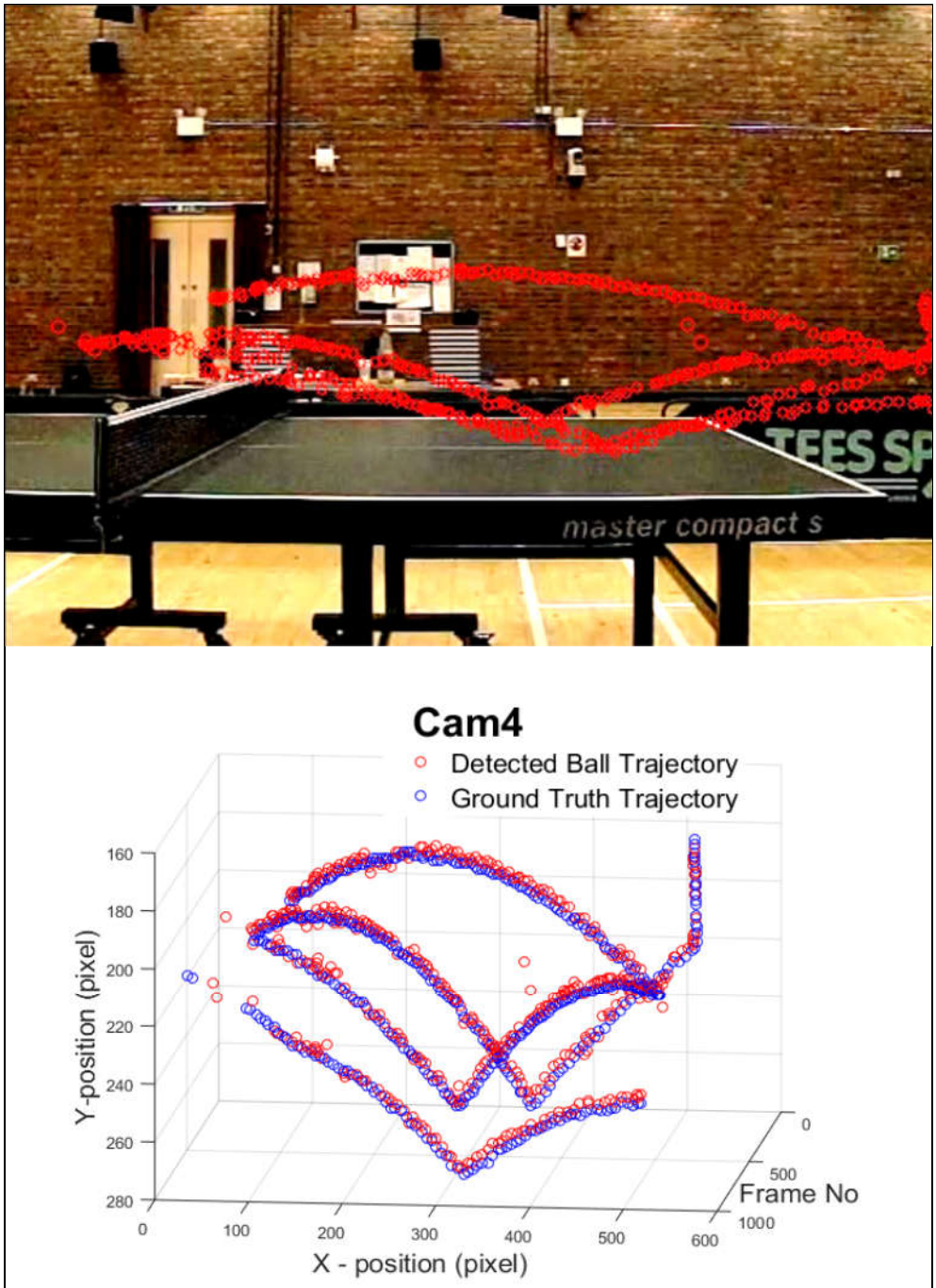
Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 2	773	91%	3.5 pixels	2.9 pixels	0.022 sec

Figure A.6: Sequence 5 - Camera 2: 2D Trajectory comparison



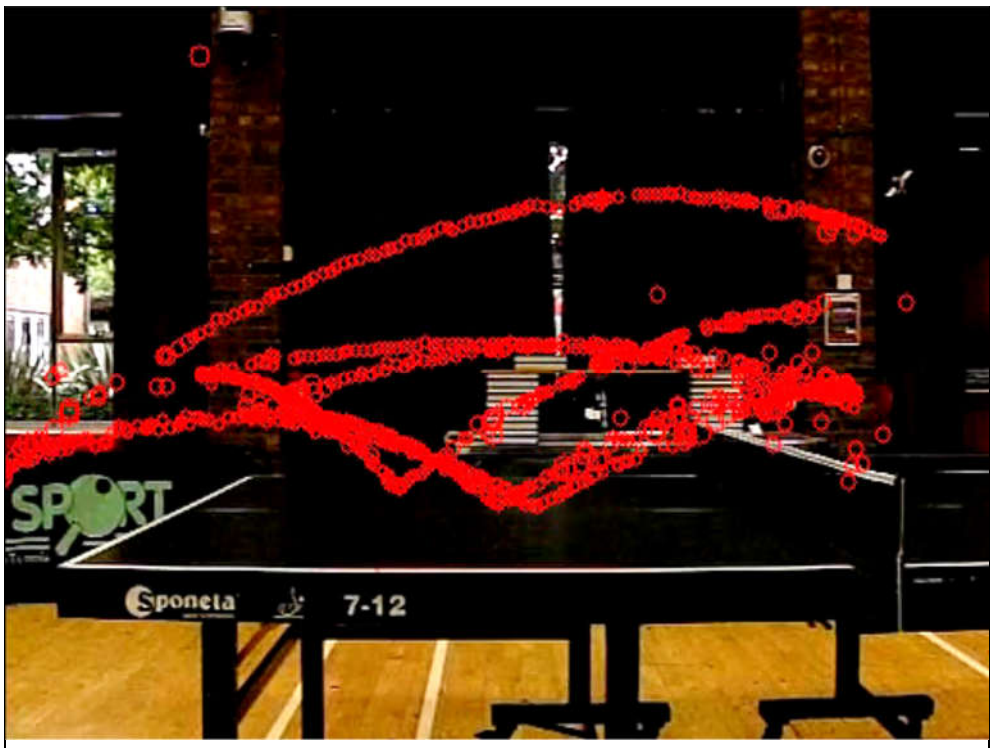
Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 3	773	96%	3.5 pixels	2.6 pixels	0.022 sec

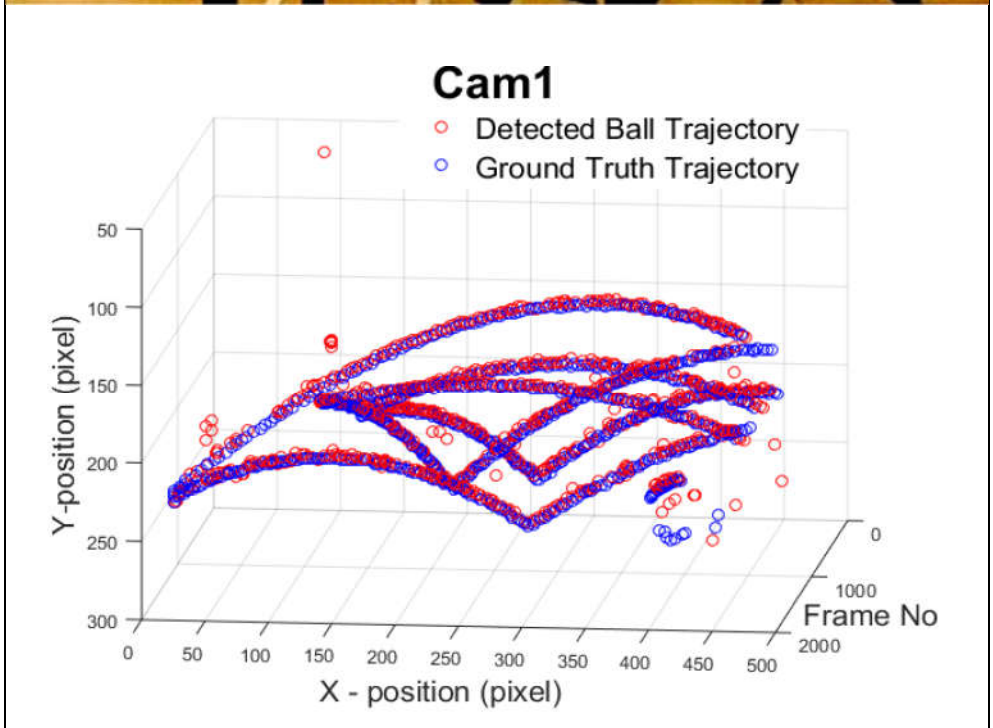
Figure A.7: Sequence 5 - Camera 3: 2D Trajectory comparison



Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 4	834	98%	3.5 pixels	3.9 pixels	0.028 sec

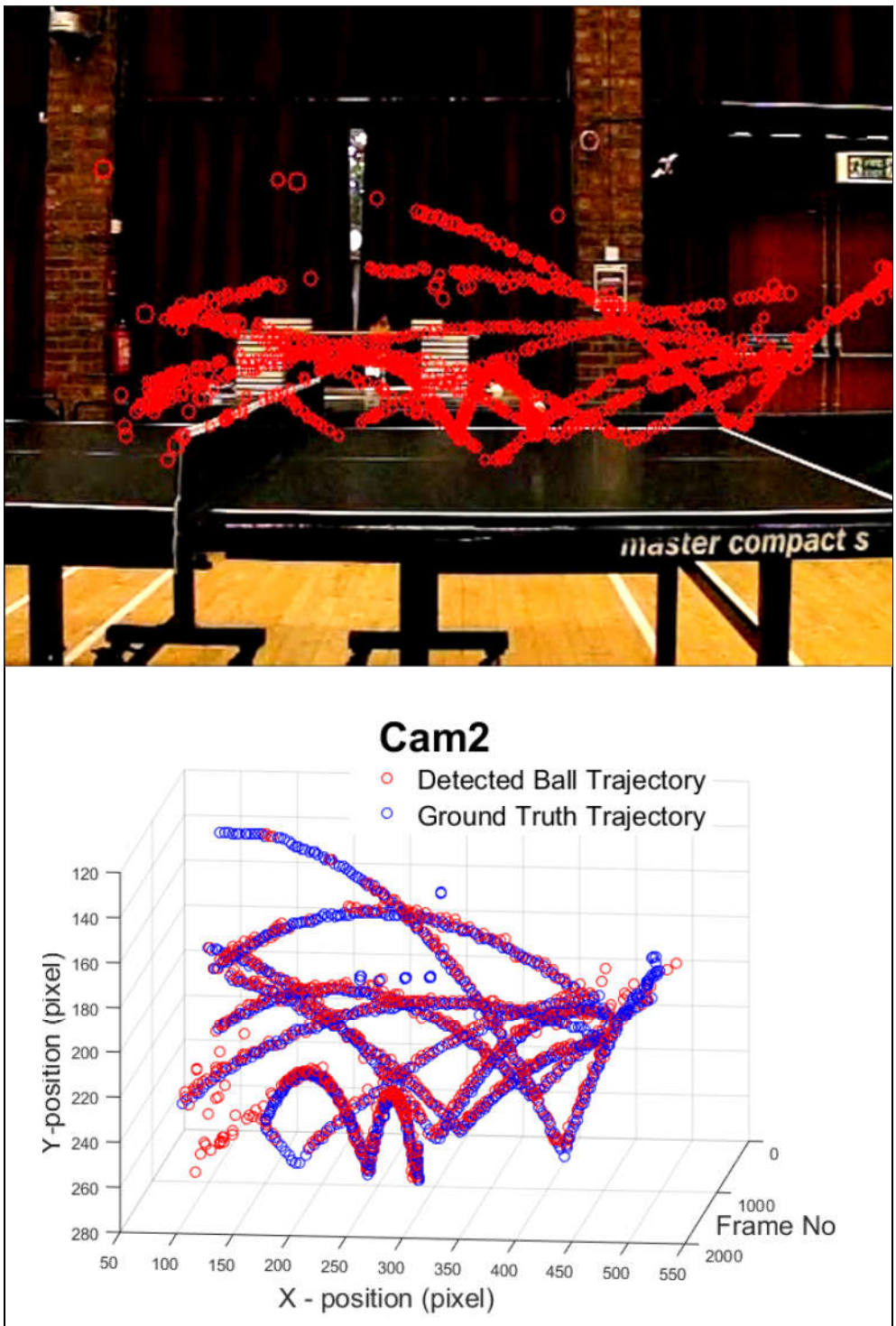
Figure A.8: Sequence 5 - Camera 4: 2D Trajectory comparison





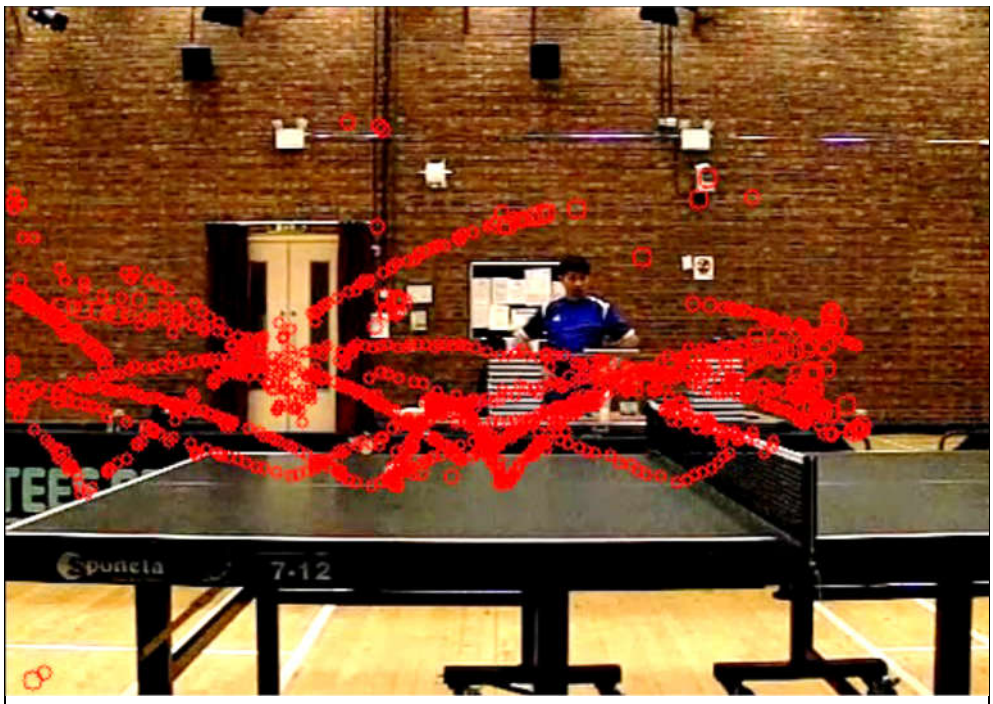
Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 1	1593	94%	3.5 pixels	1.1 pixels	0.028 sec

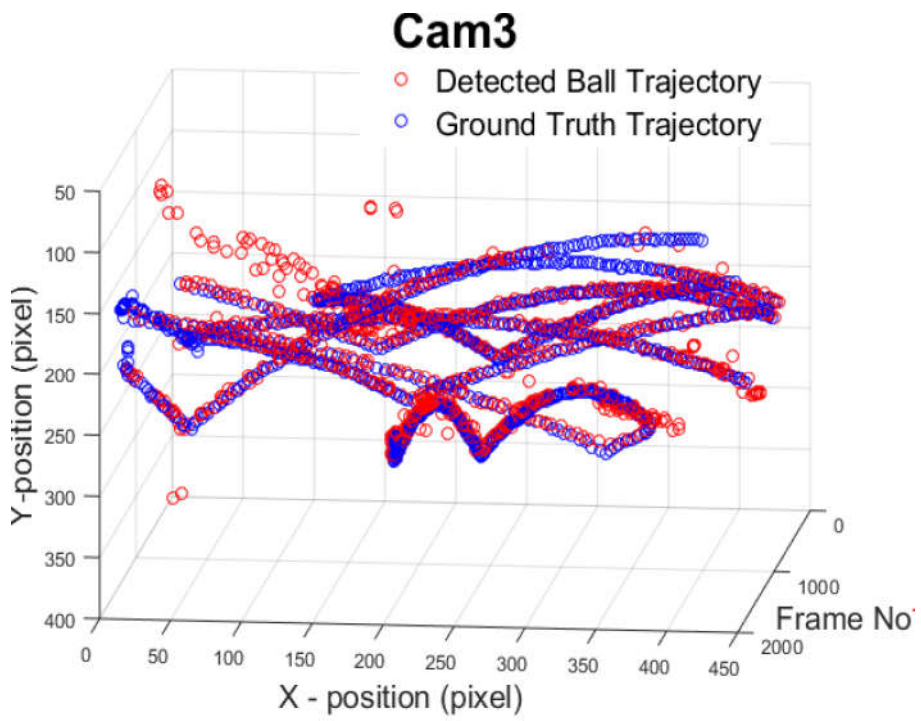
Figure A.9: Sequence 6 - Camera 1: 2D Trajectory comparison



Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 2	1797	93%	3.5 pixels	3.5 pixels	0.028 sec

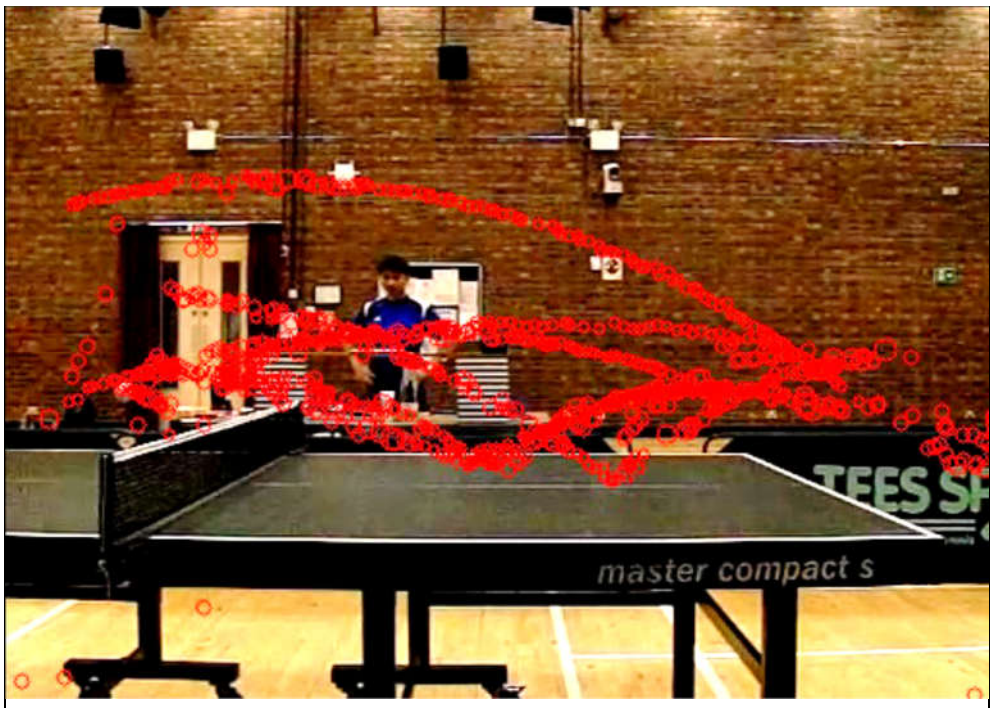
Figure A.10: Sequence 6 - Camera 2: 2D Trajectory comparison

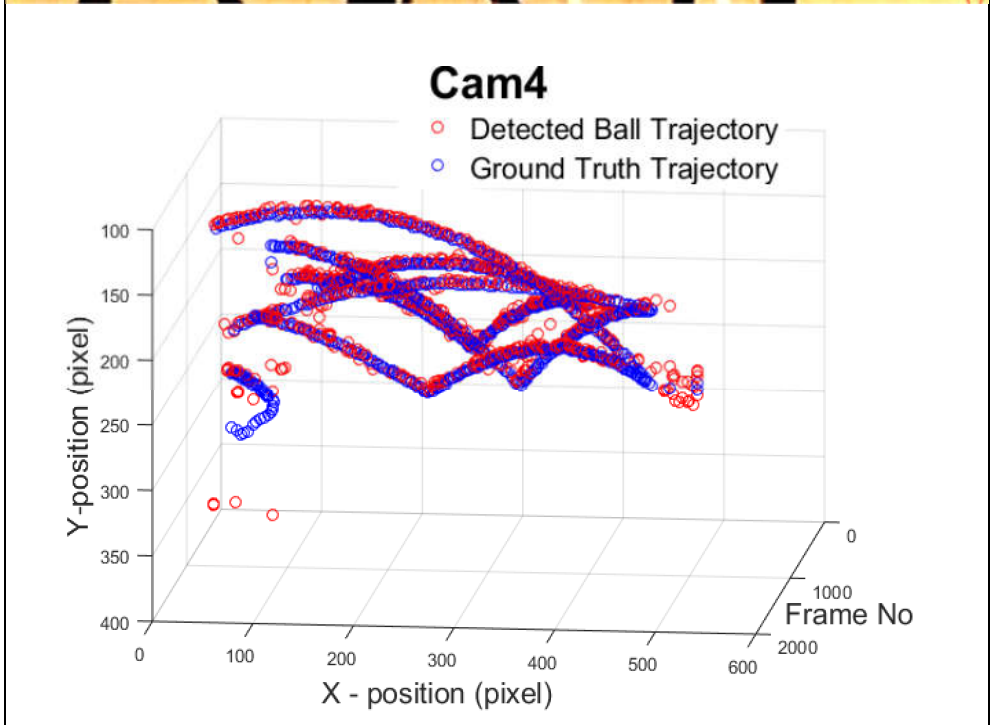




Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 3	1797	90%	3.5 pixels	2.5 pixels	0.025 sec

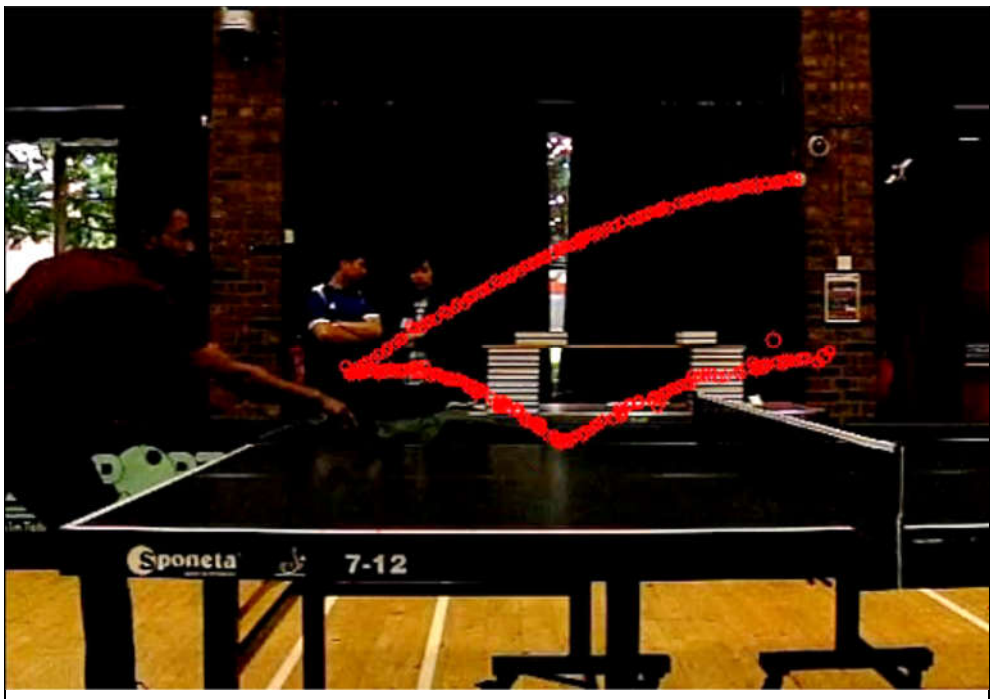
Figure A.11: Sequence 6 - Camera 3: 2D Trajectory comparison

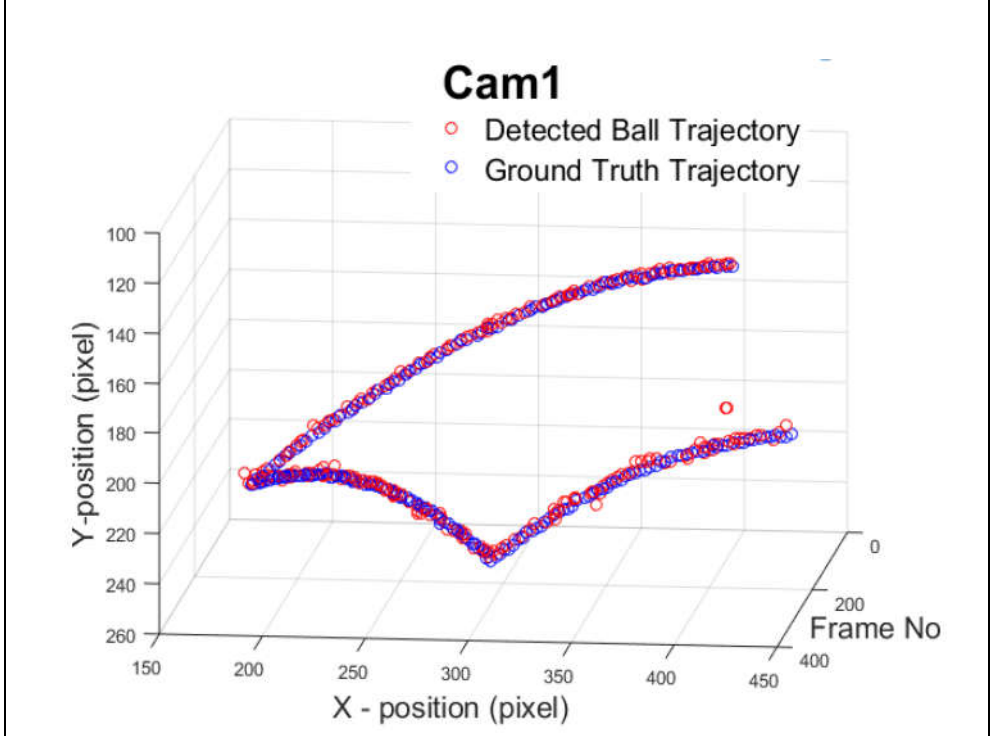




Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 4	1593	88%	3.5 pixels	1.1 pixels	0.029 sec

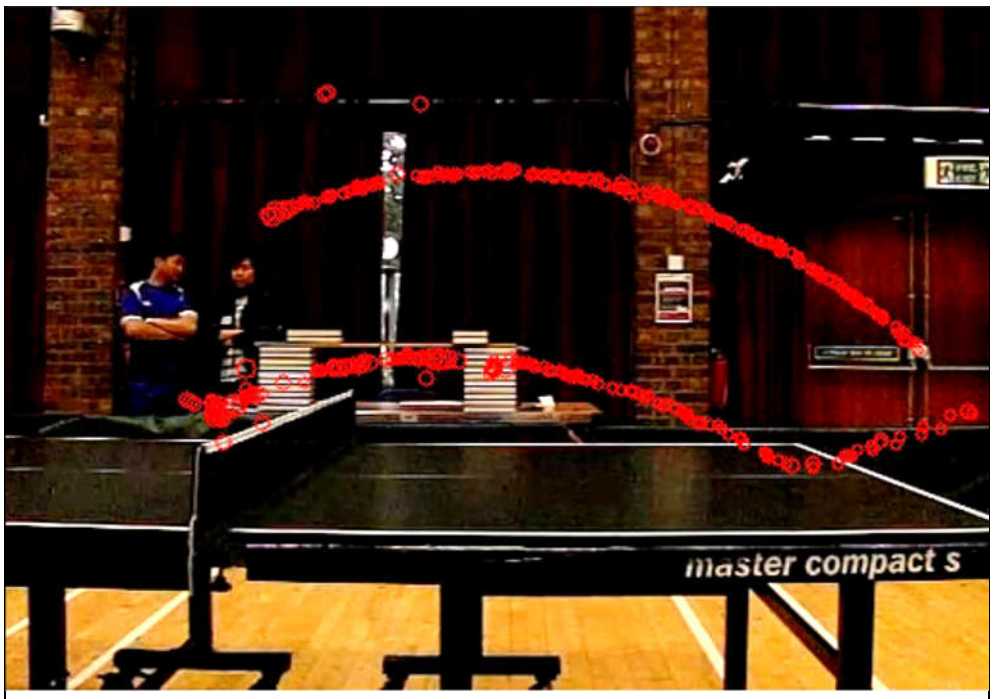
Figure A.12: Sequence 6 - Camera 4: 2D Trajectory comparison

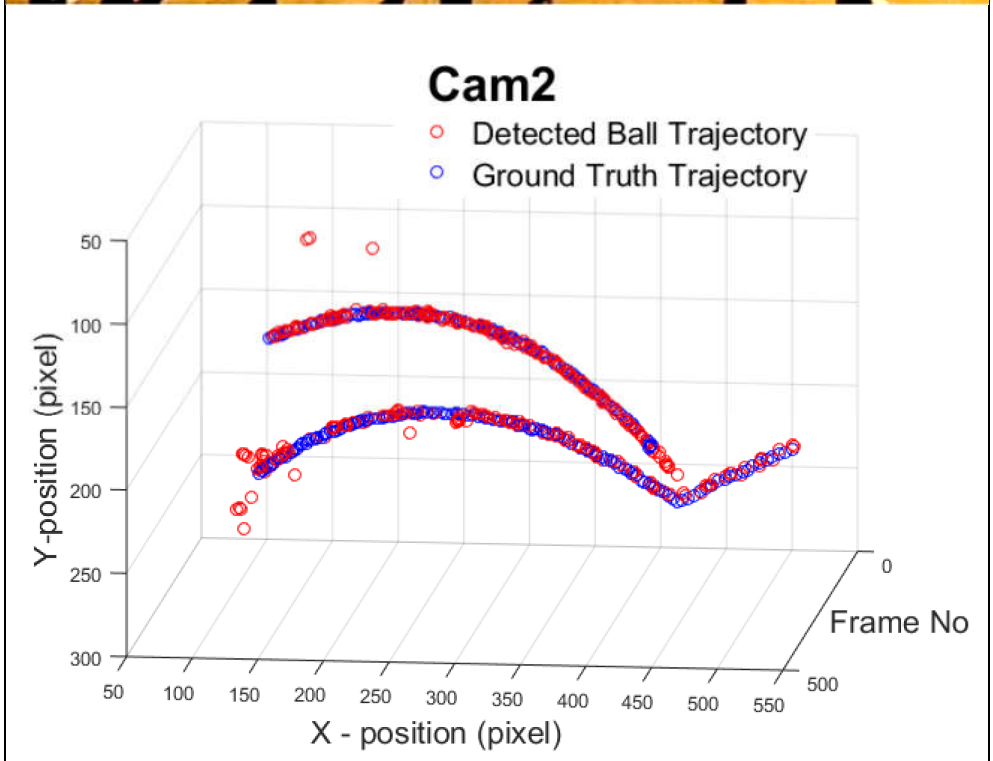




Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 1	325	99%	3.5 pixels	1.1 pixels	0.028 sec

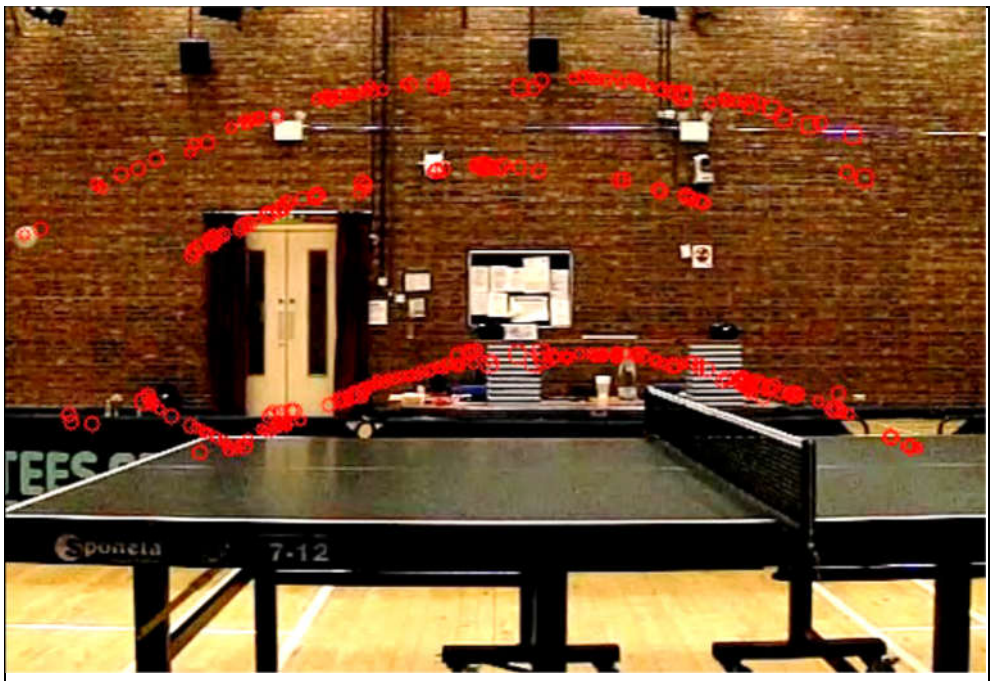
FigureA.13: Sequence 7 - Camera 1: 2D Trajectory comparison

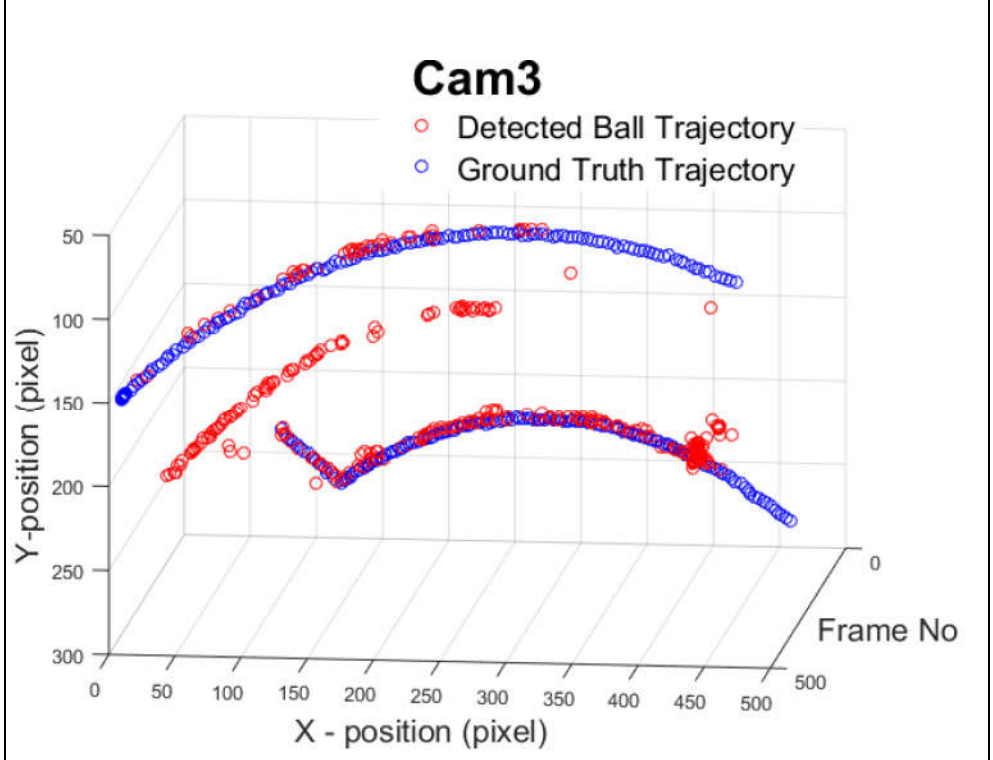




Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 2	440	87%	3.5 pixels	1.6 pixels	0.028 sec

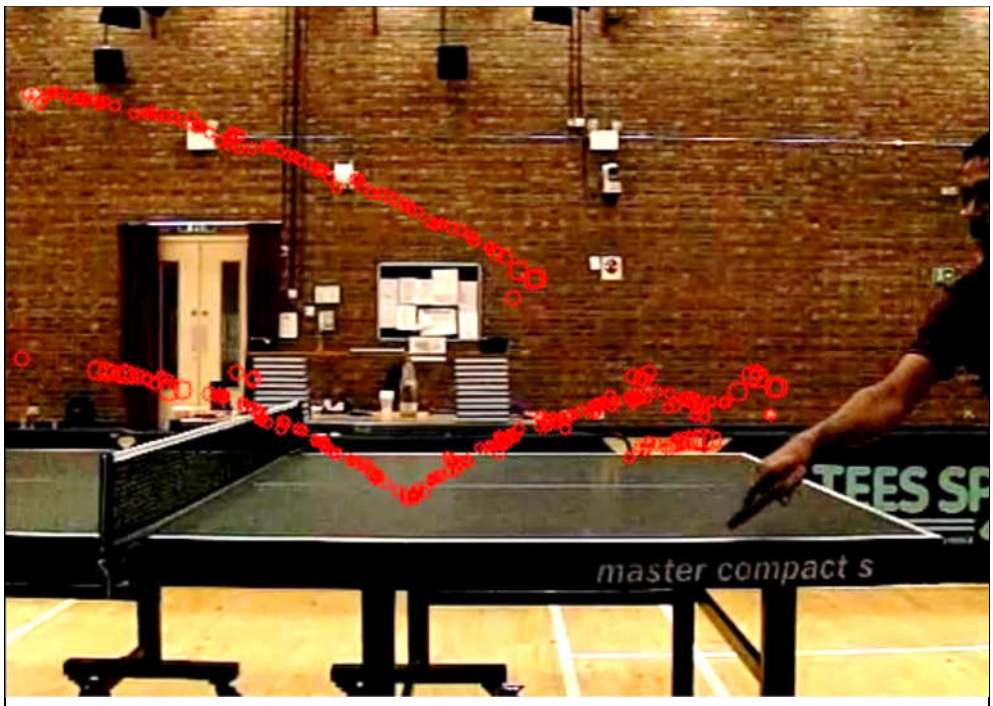
Figure A.14: Sequence 7 - Camera 2: 2D Trajectory comparison

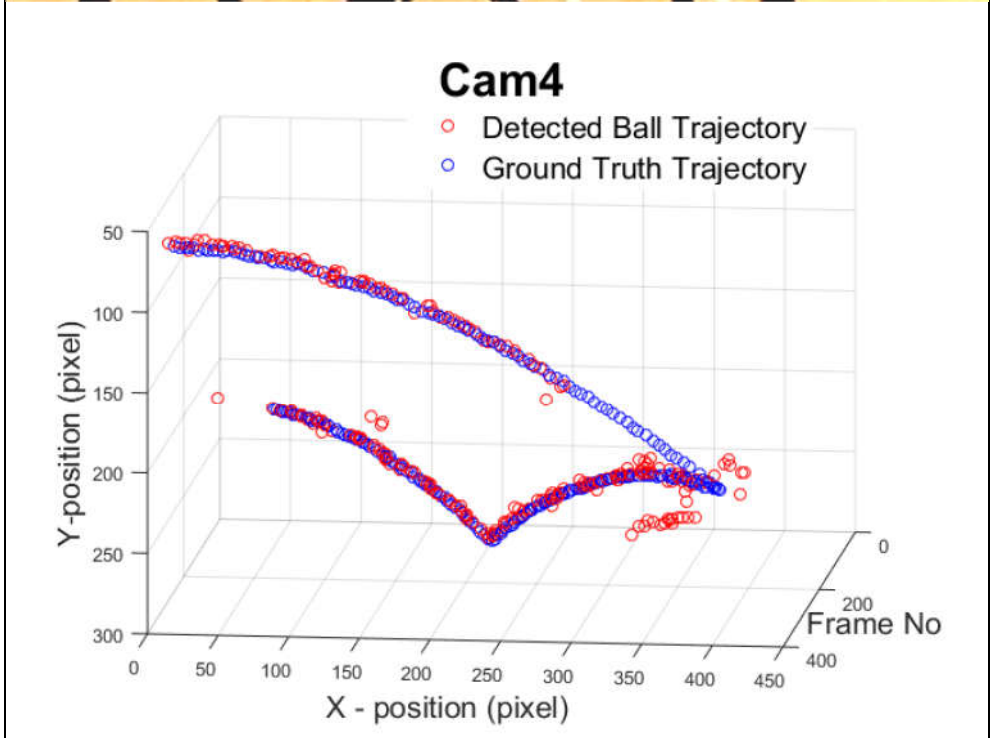




Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 3	440	75%	3.5 pixels	1.1 pixels	0.025 sec

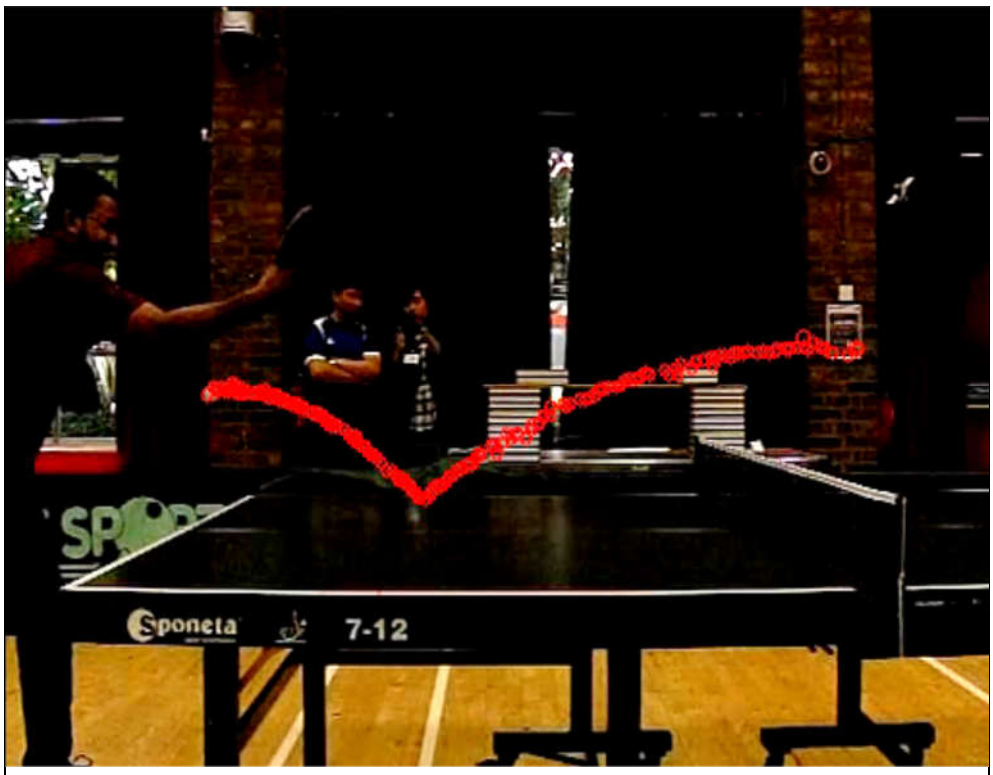
Figure A.15: Sequence 7 - Camera 3: 2D Trajectory comparison

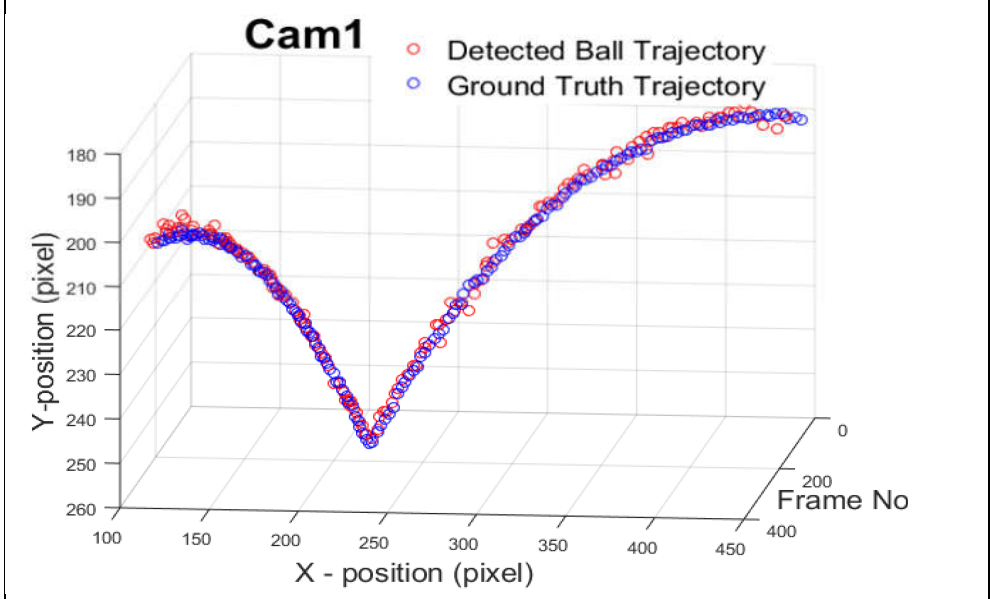




Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 4	325	89 %	3.6 pixels	3.6 pixels	0.026 sec

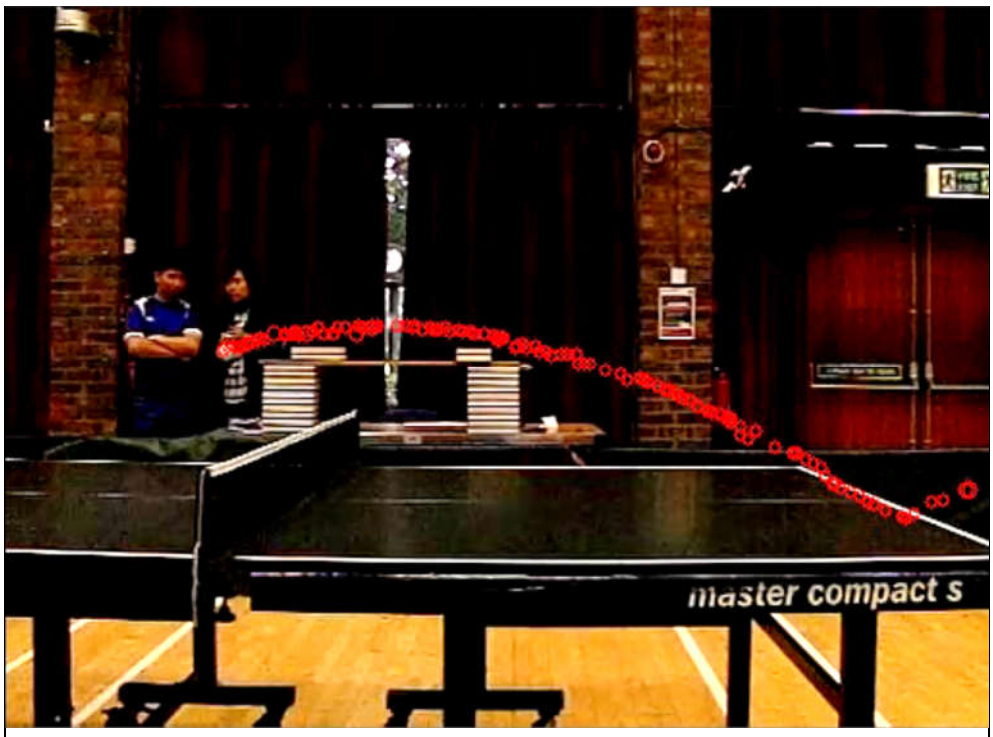
Figure A.16: Sequence 7 - Camera 4: 2D Trajectory comparison

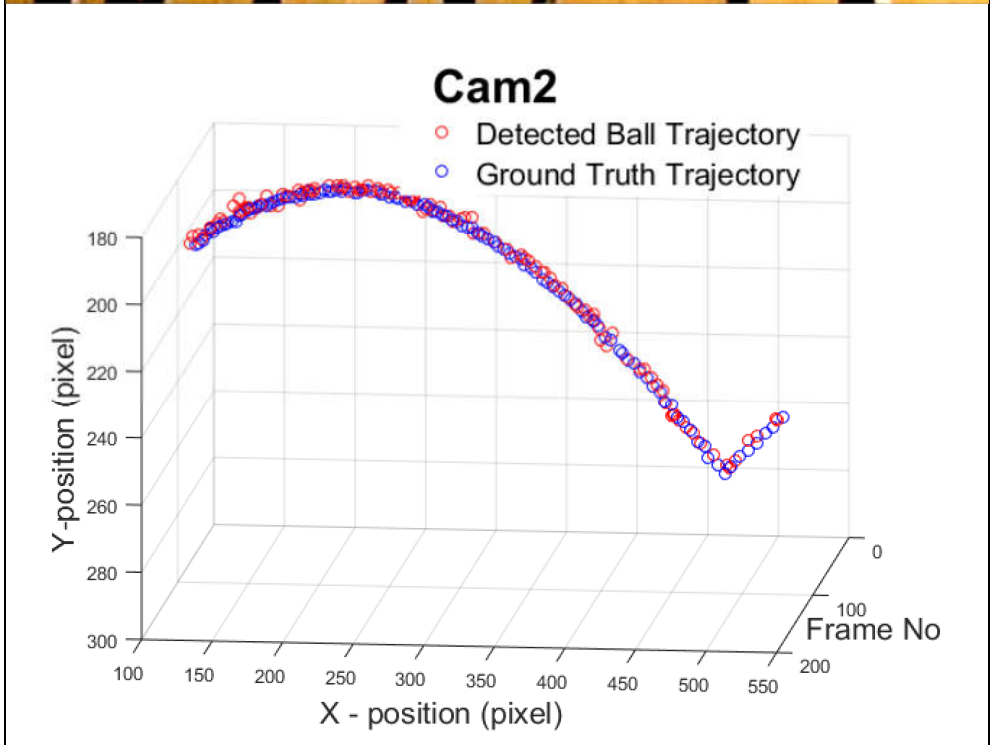




Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 1	245	98%	3.5 pixels	2.6 pixels	0.021 sec

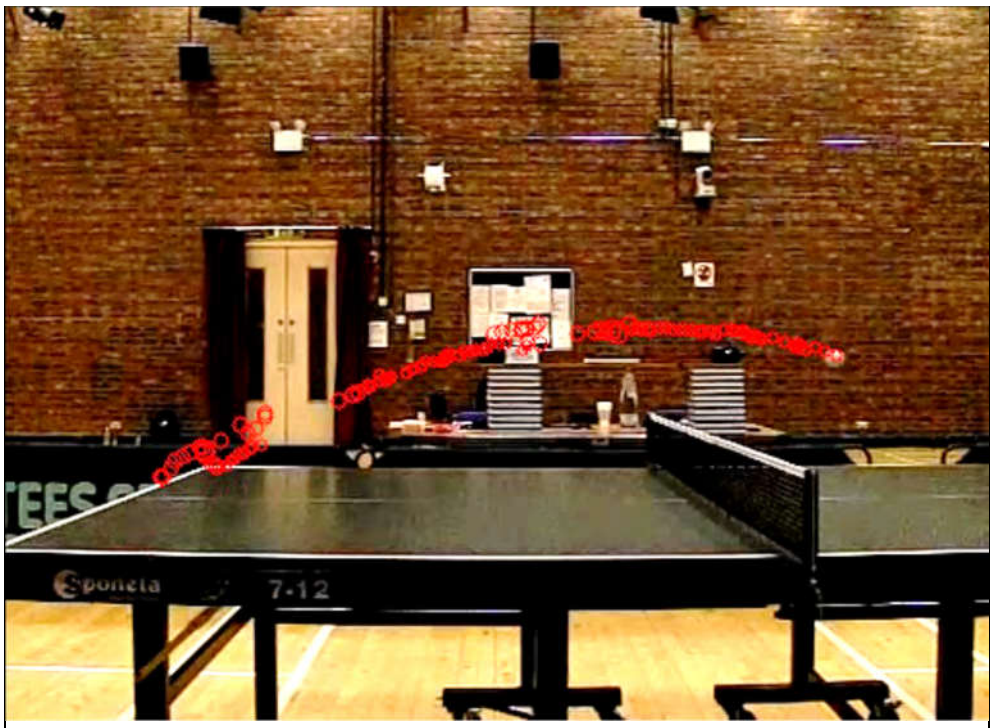
Figure A.17: Sequence 8 - Camera 1: 2D Trajectory comparison

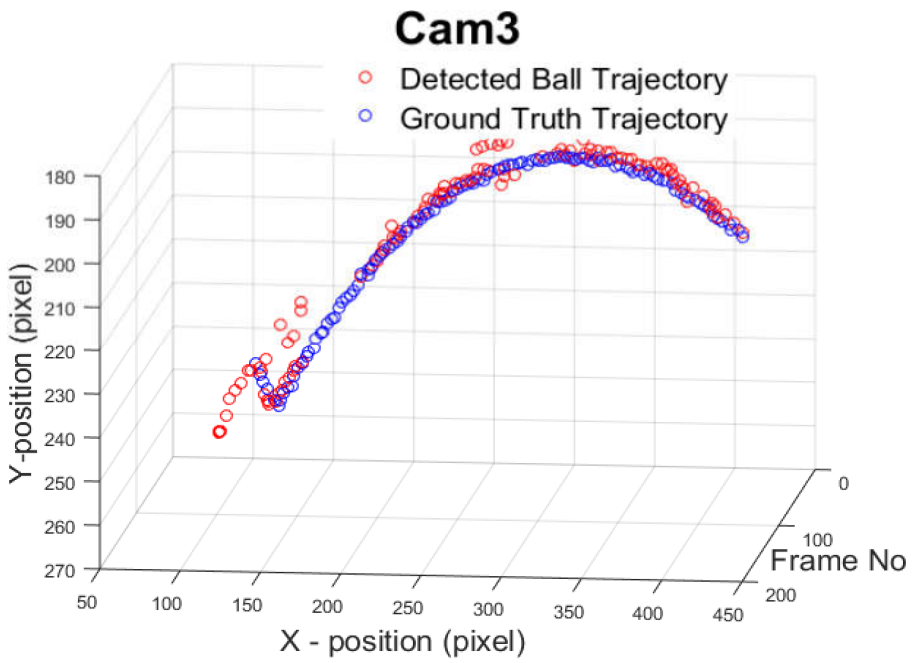




Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 2	116	97%	3.5 pixels	2.1 pixels	0.02 sec

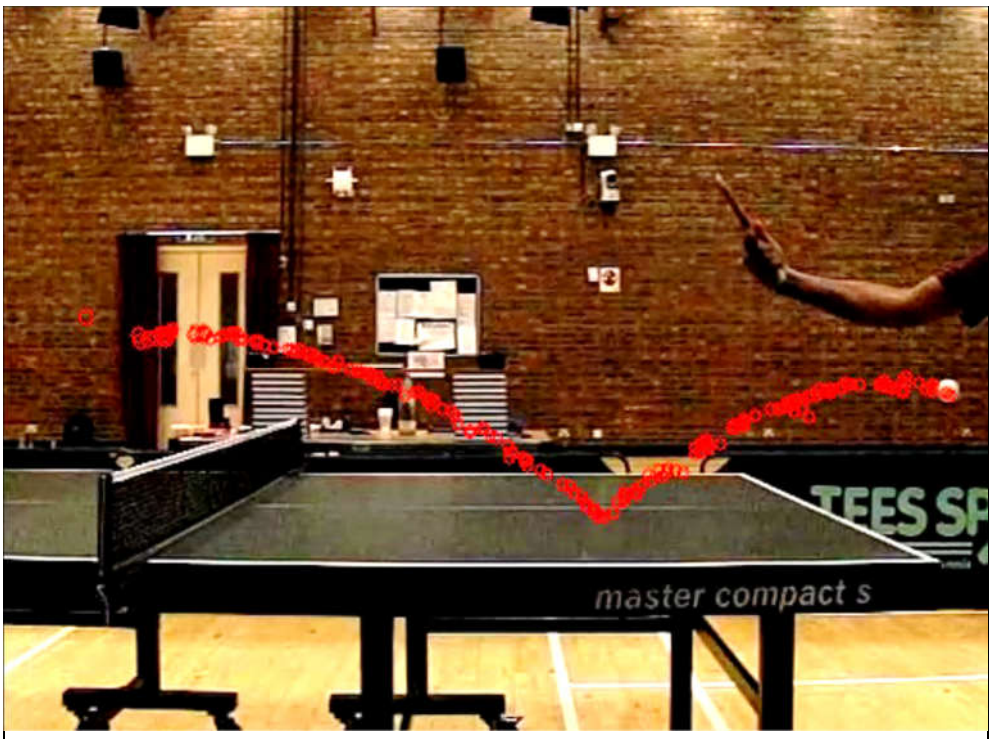
Figure A.18: Sequence 8 - Camera 2: 2D Trajectory comparison

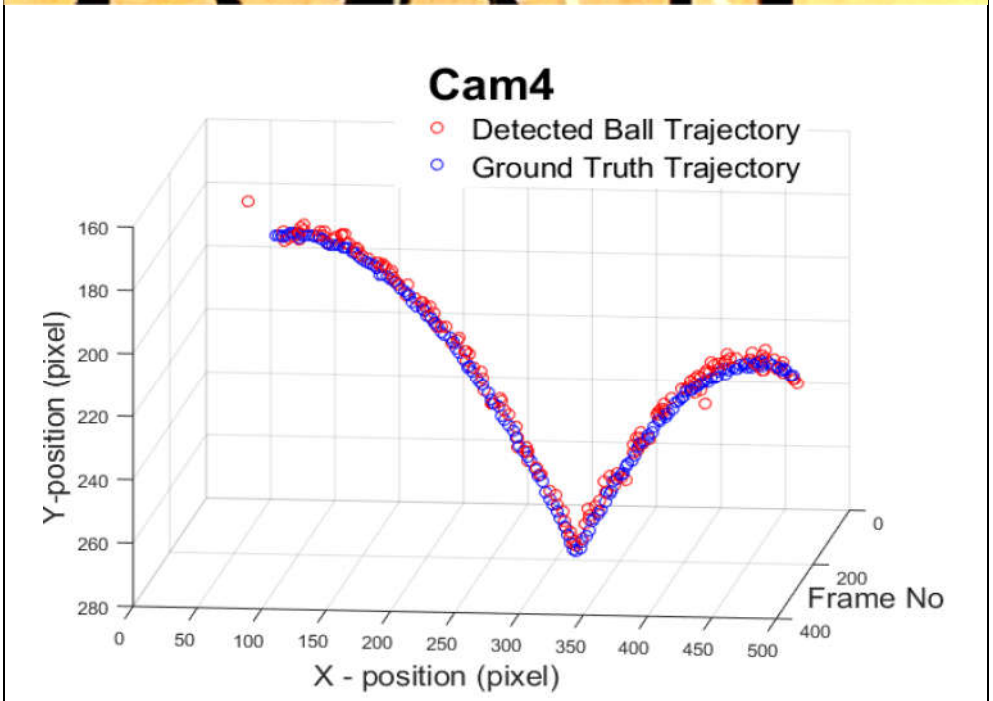




Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 3	116	79 %	3.5 pixels	3.2 pixels	0.028 sec

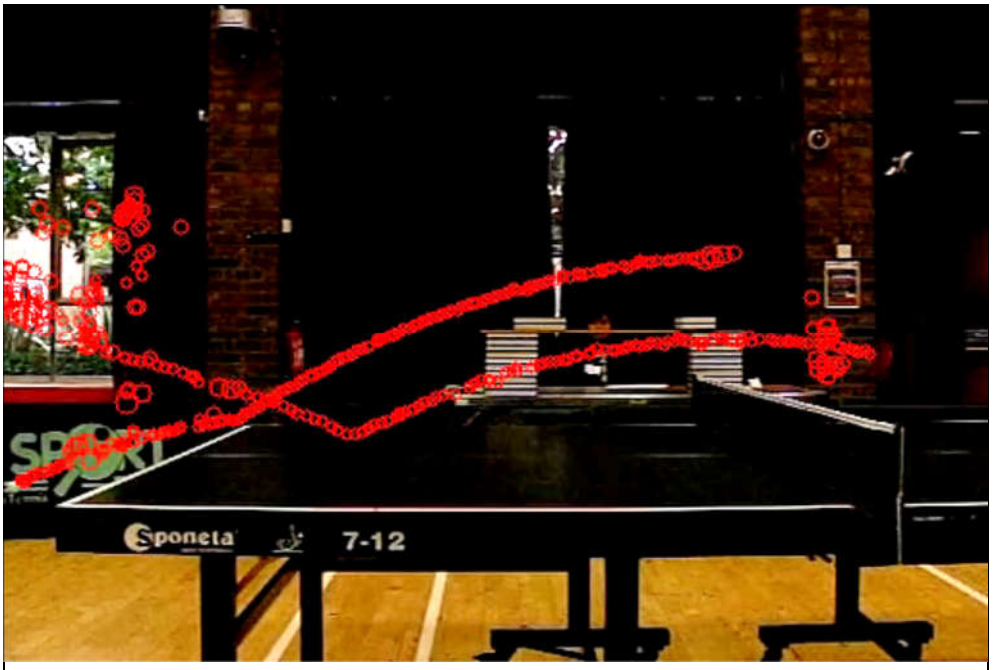
Figure A.19: Sequence 8 - Camera 3: 2D Trajectory comparison

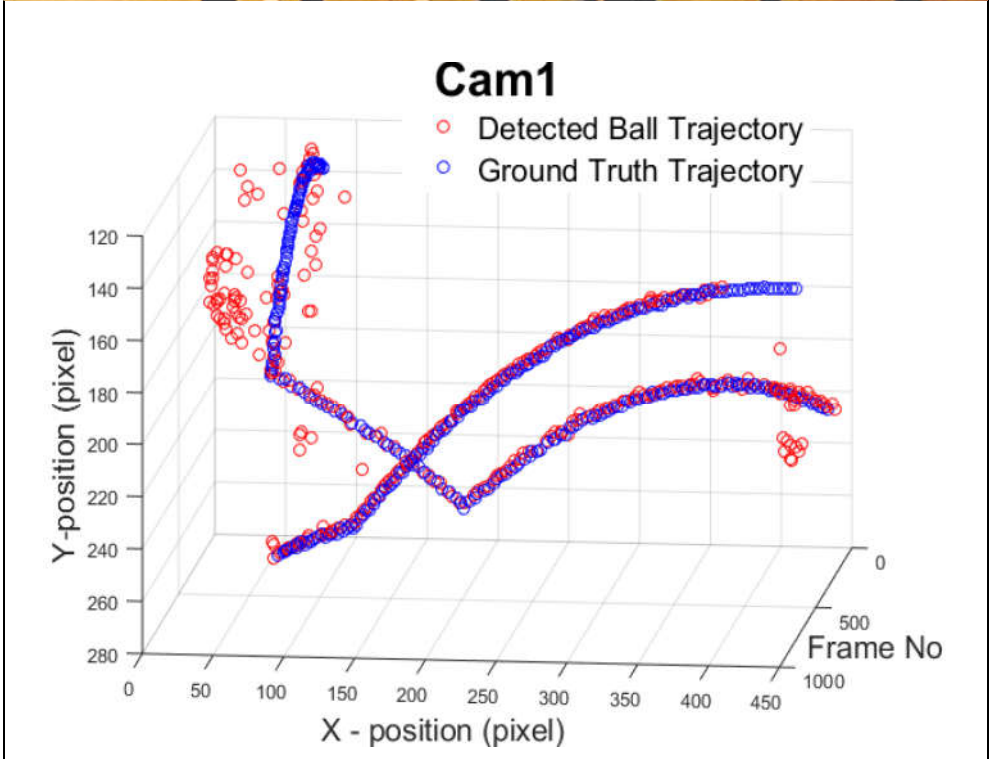




Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 4	245	99%	3.5 pixels	2.7 pixels	0.021 sec

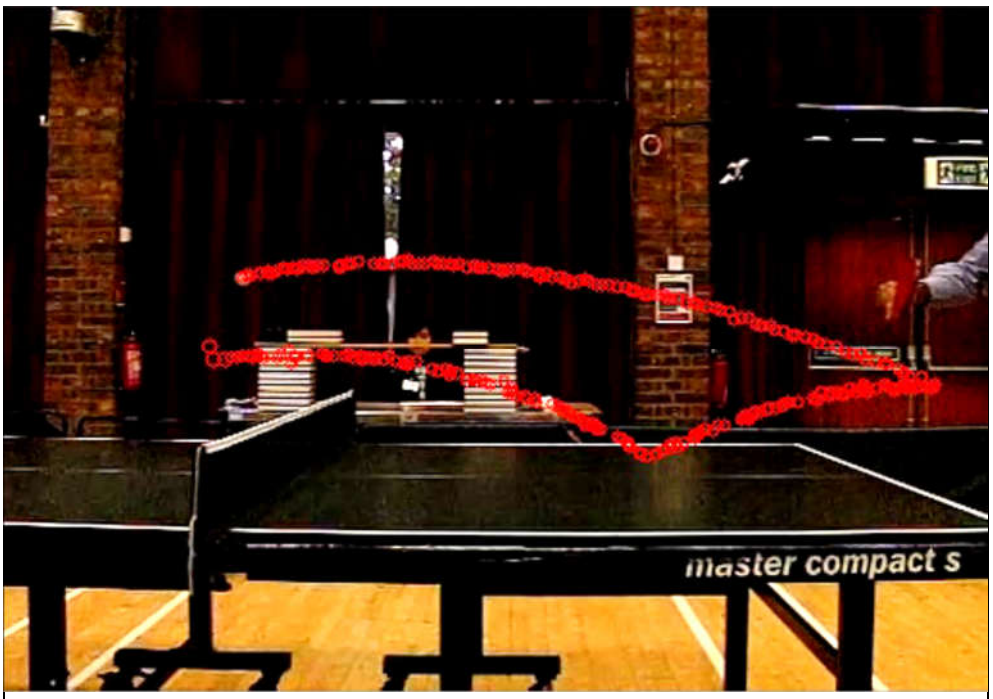
Figure A.20: Sequence 8 - Camera 4: 2D Trajectory comparison

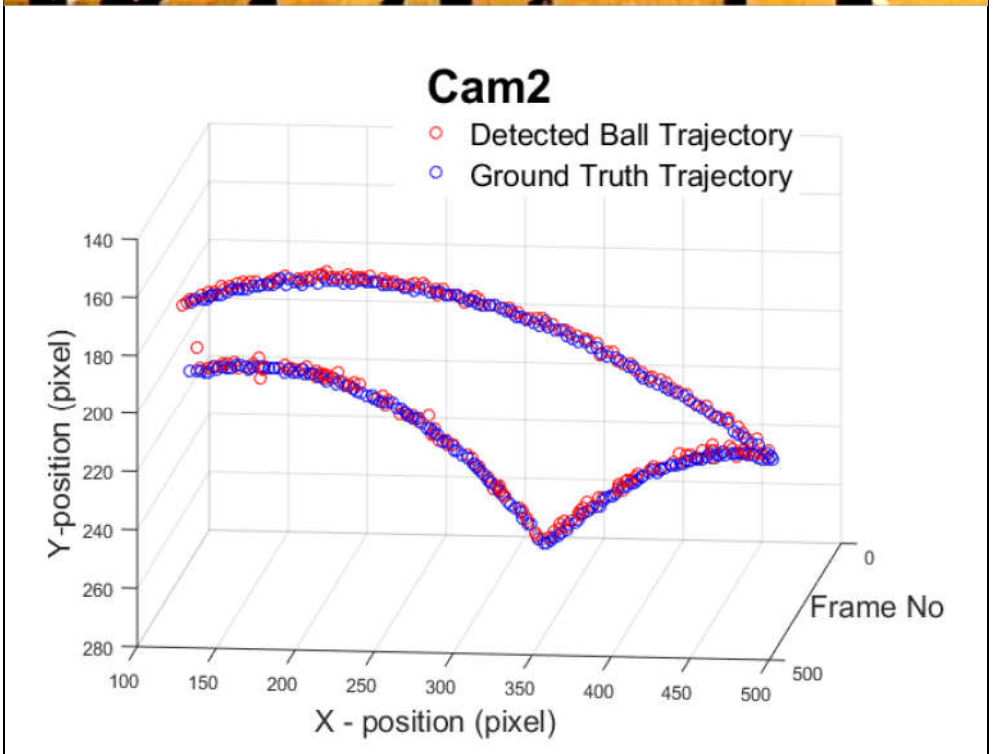




Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 1	565	84 %	3.5 pixels	3.5 pixels	0.052 sec

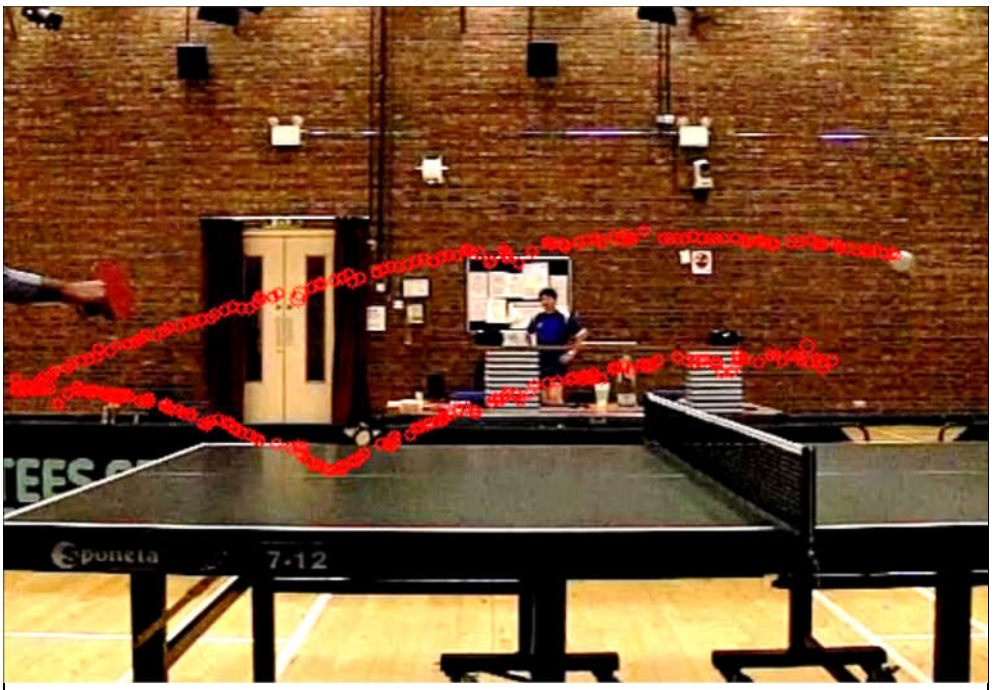
Figure A.21: Sequence 9 - Camera 1: 2D Trajectory comparison

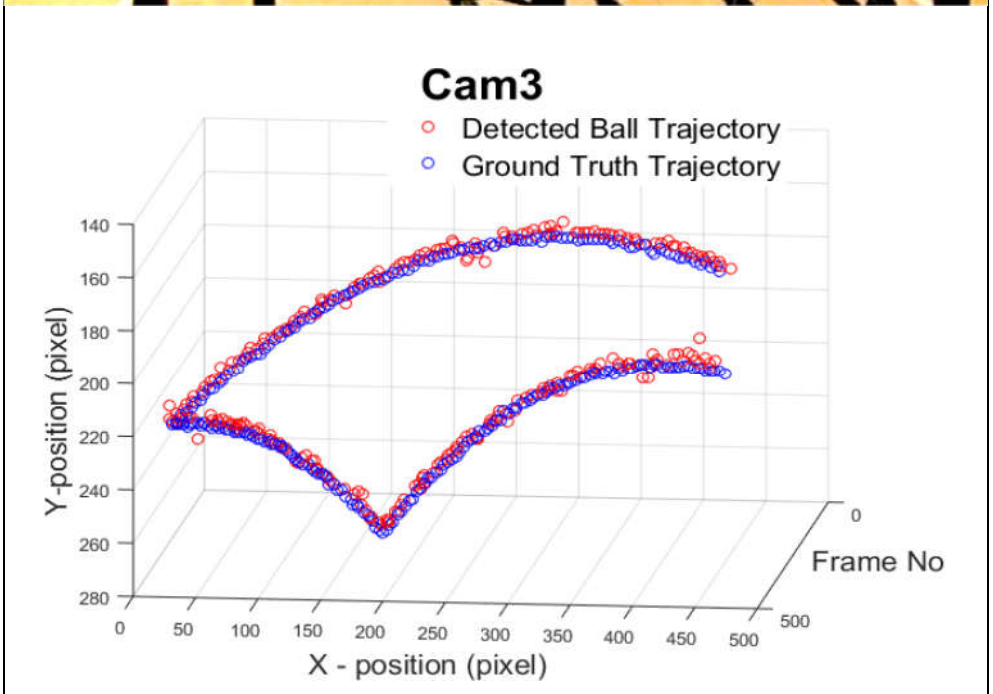




Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 2	429	98%	3.5 pixels	1.1 pixels	0.028 sec

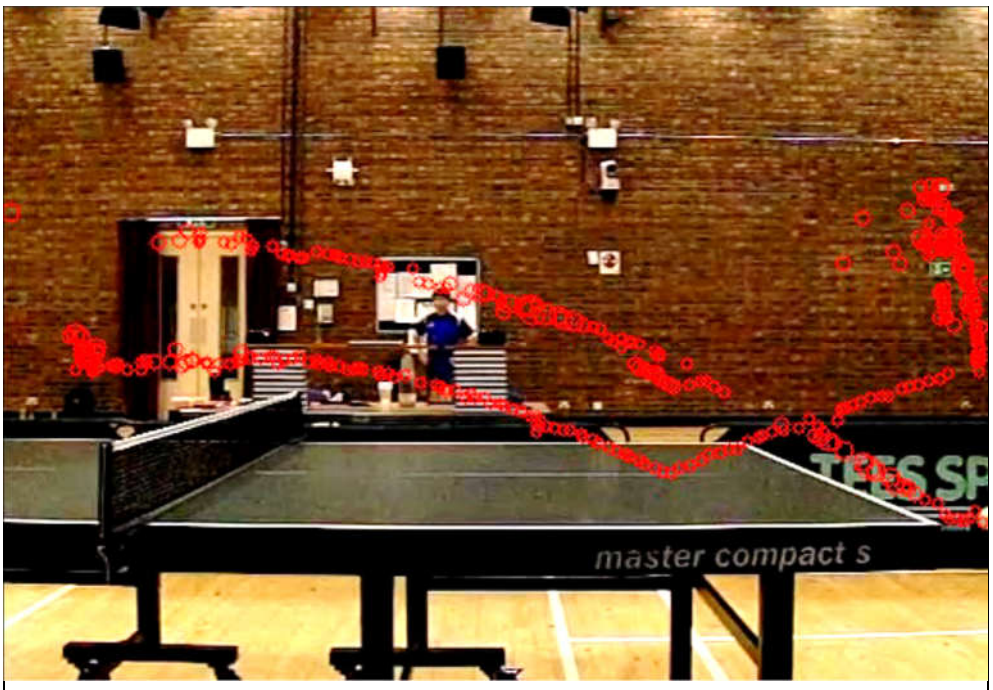
Figure A.22: Sequence 9 - Camera 2: 2D Trajectory comparison

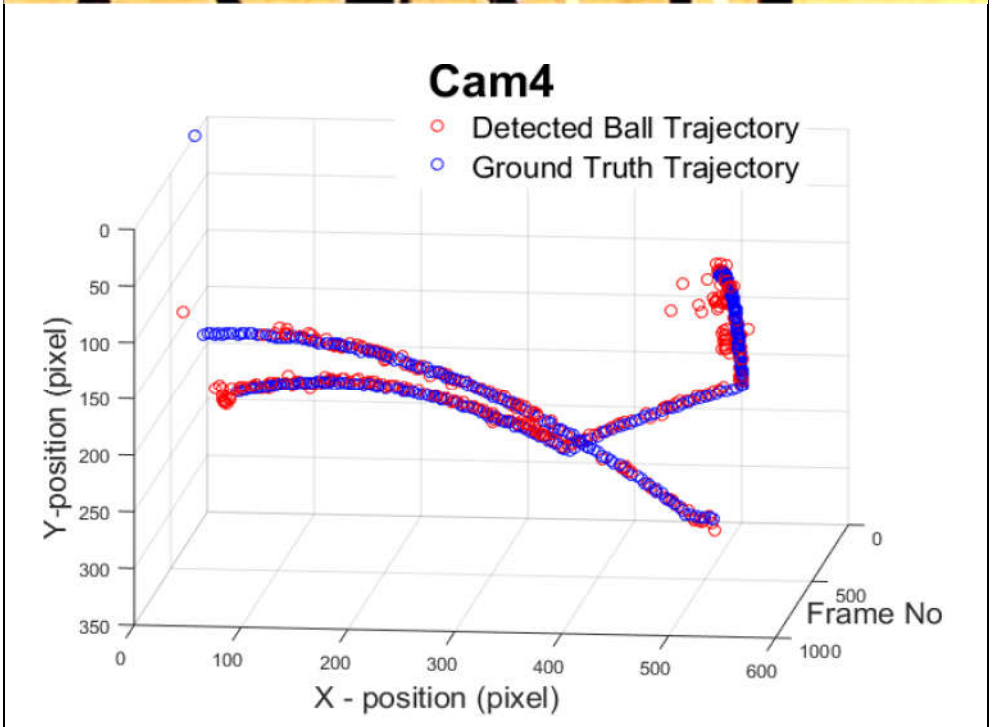




Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 3	429	89%	3.5 pixels	2.7 pixels	0.025 sec

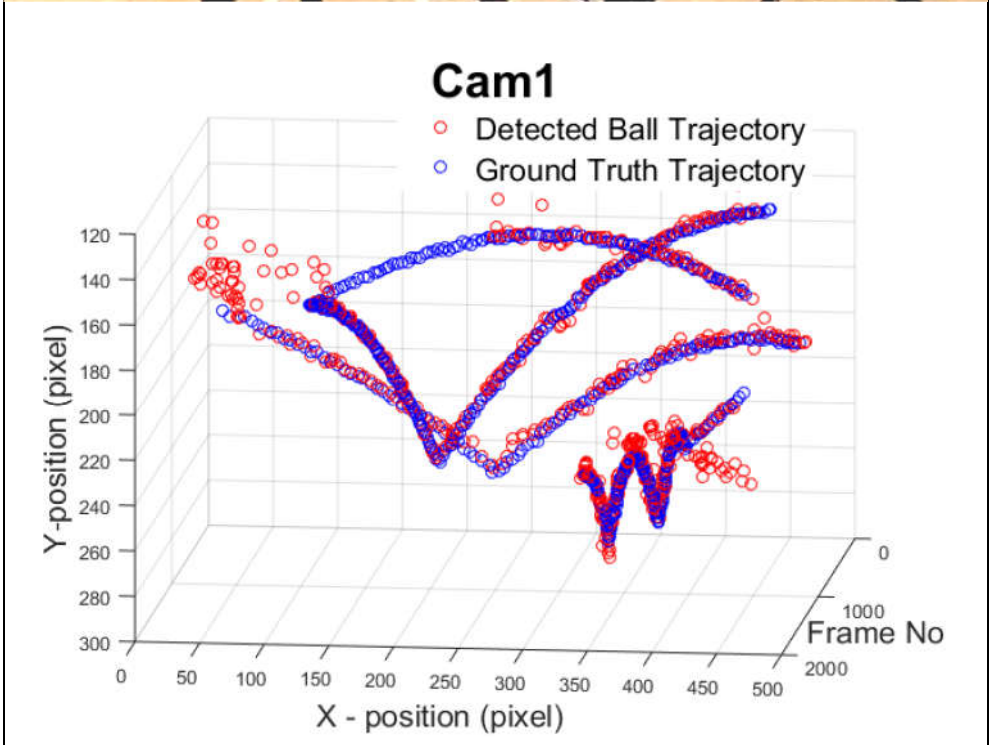
Figure A.23: Sequence 9 - Camera 3: 2D Trajectory comparison





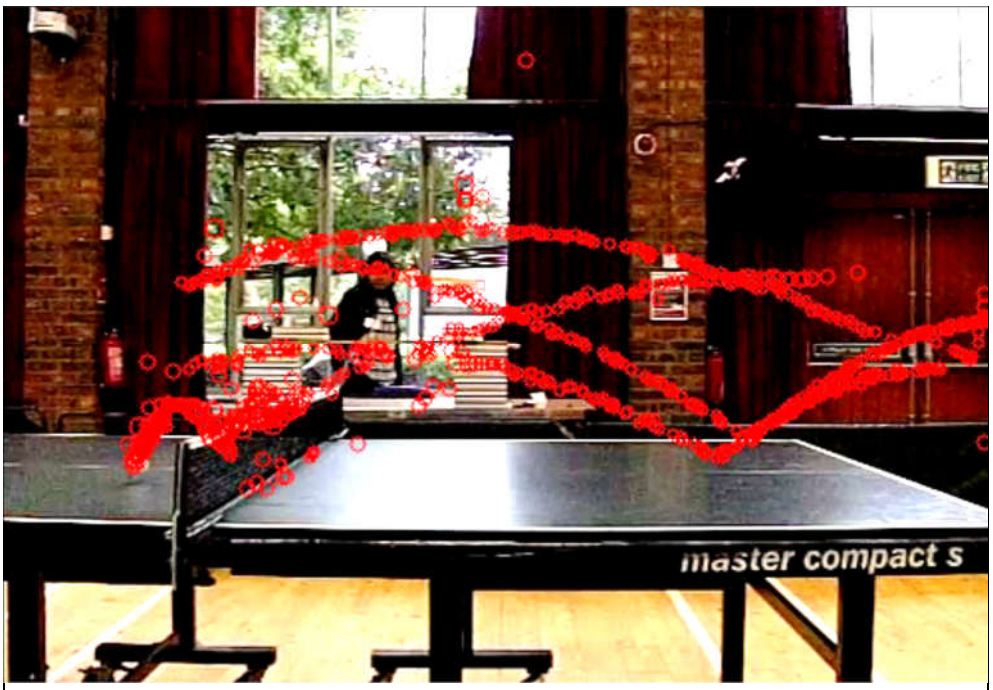
Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 4	565	79%	3.5 pixels	3 pixels	0.052 sec

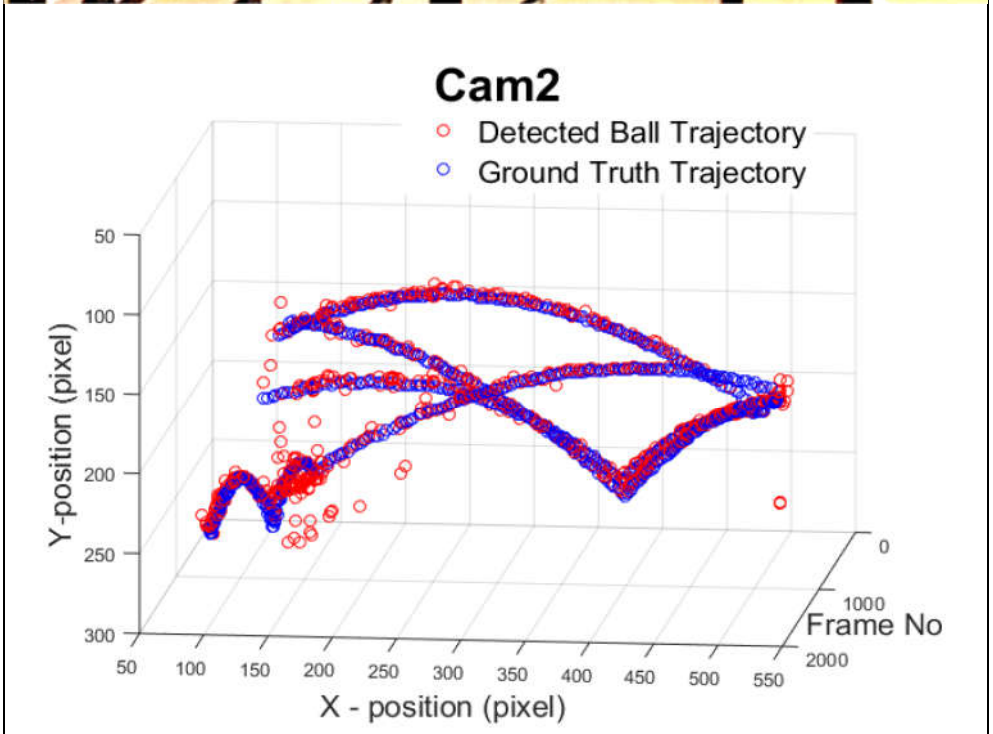
Figure A.24: Sequence 9 - Camera 4: 2D Trajectory comparison



Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 1	1146	91%	3.5 pixels	3.5 pixels	0.028 sec

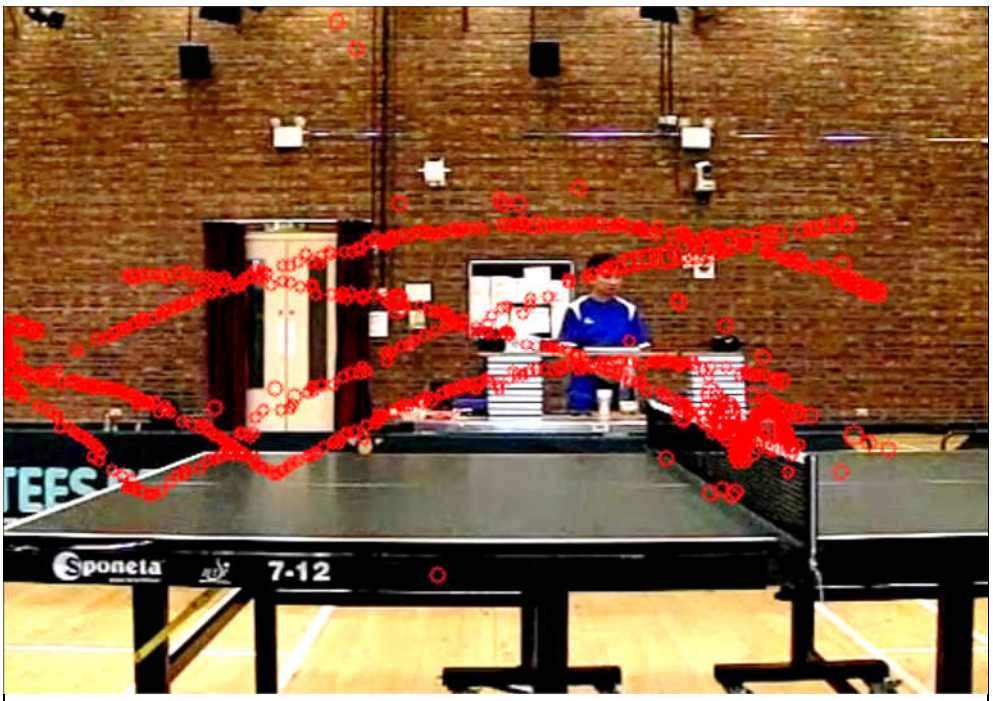
Figure A.25: Sequence 10 - Camera 1: 2D Trajectory comparison

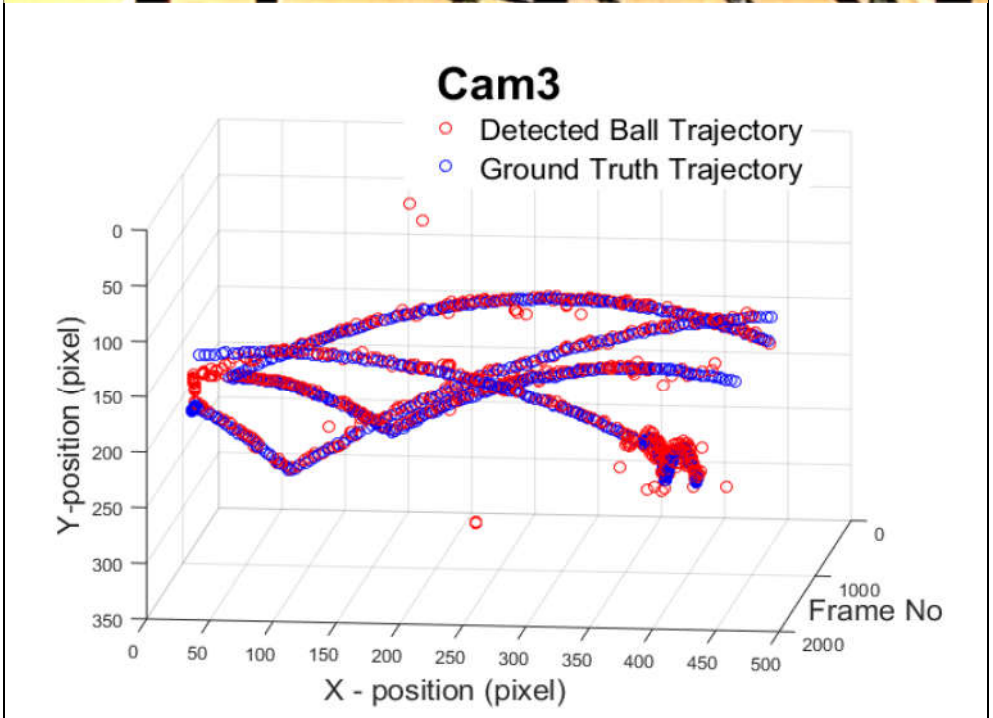




Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 2	1100	95%	3.5 pixels	3.2 pixels	0.028 sec

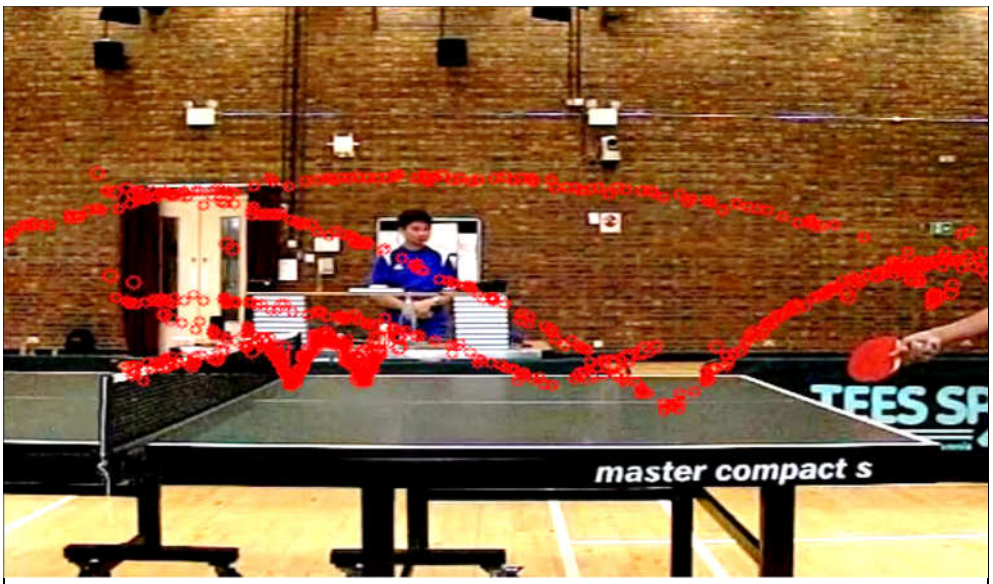
Figure A.26: Sequence 10 - Camera 2: 2D Trajectory comparison

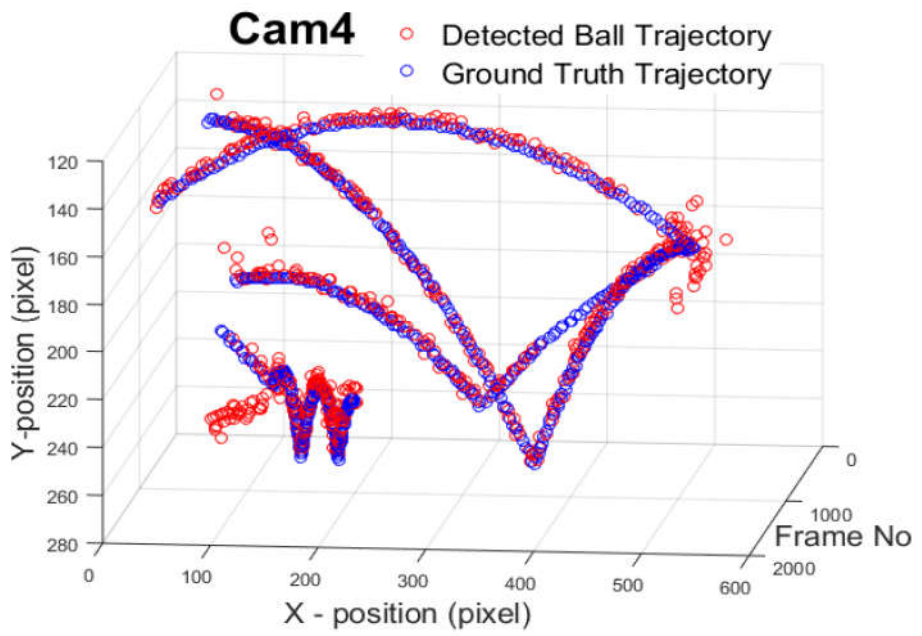




Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 3	1100	95%	3.5 pixels	1.6 pixels	0.025 sec

Figure A.27: Sequence 10 - Camera 3: 2D Trajectory comparison





Quantitative Result	Total frames	Detection rate	Average radius of ball	Average RMSE	Detection time
Camera 4	1146	86%	3.5 pixels	1.8 pixels	0.028 sec

Figure A.28: Sequence 10 - Camera 4: 2D Trajectory comparison

Reference

Abdelli, A. and Choi, H.-J. (2017) 'A four-frames differencing technique for moving objects detection in wide area surveillance', *2017 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pp. 210–214 [Online]. DOI: 10.1109/BIGCOMP.2017.7881701.

Agarwal, A., Gupta, S. and Singh, D. K. (2016) 'Review of optical flow technique for moving object detection', *2016 2nd International Conference on Contemporary Computing and Informatics (IC3I)*, pp. 409–413 [Online]. DOI: 10.1109/IC3I.2016.7917999.

Anon (2013) *All About Ping Pong Balls*, [Online]. Available at http://www.tabletennismaster.com/page/ping-pong-balls (Accessed 15 July 2013).

Anon (2017) 'Jade Site | Java Agent DEvelopment Framework', [Online]. Available at http://jade.tilab.com/ (Accessed 11 November 2017).

Anuj, L. and Krishna, M. T. G. (2017) 'Multiple camera based multiple object tracking under occlusion: A survey', *2017 International Conference on Innovative Mechanisms for Industry Applications (ICIMIA)*, pp. 432–437 [Online]. DOI: 10.1109/ICIMIA.2017.7975652.

Arenas, M., Ruiz-del-Solar, J., Norambuena, S. and Cubillos, S. (2009) 'A Robot Referee for Robot Soccer', in Iocchi, L., Matsubara, H., Weitzenfeld, A., and Zhou, C. (eds), *RoboCup 2008: Robot Soccer World Cup XII*, Lecture Notes in Computer Science, Springer Berlin Heidelberg, pp. 426–438 [Online]. Available at http://link.springer.com/chapter/10.1007/978-3-642-02921-9_37 (Accessed 22 September 2013).

Bacic, B., Meng, Q. and Chan, K. Y. (2017) 'Privacy Preservation for eSports: A Case Study Towards Augmented Video Golf Coaching System', *2017 10th International Conference on Developments in eSystems Engineering (DeSE)*, pp. 169–174 [Online]. DOI: 10.1109/DeSE.2017.34.

Bal, B. and Dureja, G. (2012) 'Hawk Eye: A Logical Innovative Technology Use in Sports for Effective Decision Making', *Sport Science Review*, vol. XXI, no. 1, pp. 107–119 [Online]. DOI: 10.2478/v10237-012-0006-6.

Balaji, S. R. and Karthikeyan, S. (2017) 'A survey on moving object tracking using image processing', *2017 11th International Conference on Intelligent Systems and Control (ISCO)*, pp. 469–474 [Online]. DOI: 10.1109/ISCO.2017.7856037.

Bao, H., Chen, X., Wang, Z., Pan, M. and Meng, F. (2012) 'Bouncing model for the table tennis trajectory prediction and the strategy of hitting the ball', *2012 International Conference on Mechatronics and Automation (ICMA)*, pp. 2002–2006 [Online]. DOI: 10.1109/ICMA.2012.6285129.

Bayona, A., SanMiguel, J. C. and Martinez, J. M. (2010) 'Stationary foreground detection using background subtraction and temporal difference in video

surveillance', *2010 17th IEEE International Conference on Image Processing (ICIP)*, pp. 4657–4660 [Online]. DOI: 10.1109/ICIP.2010.5650699.

BBC (2018) 'World Cup to use goal-line system', *BBC*, 19th February [Online]. Available at http://www.bbc.co.uk/sport/0/football/21505488 (Accessed 29 January 2018).

Ben-Ari, M. (2006) *Principles of Concurrent and Distributed Programming*, Pearson Education.

Bradski, G. and Kaehler, A. (2008a) *Learning OpenCV: Computer Vision with the OpenCV Library*, O'Reilly Media, Inc.

Bradski, G. and Kaehler, A. (2008b) *Learning OpenCV: Computer Vision with the OpenCV Library*, 1st edn, O'Reilly Media.

Brezeale, D. and Cook, D. J. (2008) 'Automatic Video Classification: A Survey of the Literature', *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 38, no. 3, pp. 416–430 [Online]. DOI: 10.1109/TSMCC.2008.919173.

Bu, J., Lao, S. Y., Bai, L., Tollari, S. and Marsala, C. (2009) 'Goalmouth Detection in Field-Ball Game Video Using Fuzzy Decision Tree', *2009 Fifth International Conference on Image and Graphics*, pp. 917–921 [Online]. DOI: 10.1109/ICIG.2009.103.

Byrd, G. (2015) '21st Century Pong', *IEEE*, vol. 48, no. 10, pp. 80–84 [Online]. DOI: 10.1109/MC.2015.306.

Cañizares, P. C., Merayo, M. G., Núñez, M. and Suárez-Paniagua, V. (2017) 'A multi-agent system architecture for statistics managing and soccer forecasting', *2017 2nd IEEE International Conference on Computational Intelligence and Applications (ICCIA)*, pp. 572–576 [Online]. DOI: 10.1109/CIAPP.2017.8167282.

Canny, J. (1986) 'A Computational Approach to Edge Detection', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698 [Online]. DOI: 10.1109/TPAMI.1986.4767851.

Casio Exilim Pro EX-F1 Sensor Info & Specs (n.d.) *Casio Exilim Pro EX-F1 Sensor Info & Specs* [Online]. Available at https://www.digicamdb.com/specs/casio_exilim-pro-ex-f1/ (Accessed 11 January 2019).

Caudill, M. (1987) 'Neural Networks Primer, Part I', *AI Expert*, vol. 2, no. 12, pp. 46–52.

Chacon-Murguia, M. I., Orozco-Rodríguez, H. E. and Ramirez-Quintana, J. A. (2017) 'Self-Adapting Fuzzy Model for Dynamic Object Detection Using RGB-D Information', *IEEE Sensors Journal*, vol. PP, no. 99, pp. 1–1 [Online]. DOI: 10.1109/JSEN.2017.2763748.

Chakroun, M., Wali, A. and Alimi, A. M. (2011) 'Multi-agent system for moving object segmentation and tracking', *2011 8th IEEE International Conference on*

Advanced Video and Signal-Based Surveillance (AVSS), pp. 424–429 [Online]. DOI: 10.1109/AVSS.2011.6027366.

Chao, W. and Jun, X. M. (2008) 'Multi-agent Based Distributed Video Surveillance System over IP', *International Symposium on Computer Science and Computational Technology, 2008. ISCSCT '08*, vol. 2, pp. 97–100 [Online]. DOI: 10.1109/ISCSCT.2008.118.

Chaple, G. N., Daruwala, R. D. and Gofane, M. S. (2015) 'Comparisions of Robert, Prewitt, Sobel operator based edge detection methods for real time uses on FPGA', *2015 International Conference on Technologies for Sustainable Development (ICTSD)*, pp. 1–4 [Online]. DOI: 10.1109/ICTSD.2015.7095920.

Charaf, M. E. H., Benattou, M. and Azzouzi, S. (2012) 'A rule-based multi-agent system for testing distributed applications', *2012 International Conference on Multimedia Computing and Systems (ICMCS)*, pp. 967–972 [Online]. DOI: 10.1109/ICMCS.2012.6320205.

Chen, L., Tsang, I. W. and Xu, D. (2012) 'Laplacian Embedded Regression for Scalable Manifold Regularization', *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 6, pp. 902–915 [Online]. DOI: 10.1109/TNNLS.2012.2190420.

Chen, W. and Zhang, Y.-J. (2006) 'Tracking Ball and Players with Applications to Highlight Ranking of Broadcasting Table Tennis Video', *IMACS Multiconference on Computational Engineering in Systems Applications*, pp. 1896–1903 [Online]. DOI: 10.1109/CESA.2006.313623.

Chen, X., Huang, Q., Zhang, W., Yu, Z., Li, R. and Lv, P. (2011) 'Ping-pong trajectory perception and prediction by a PC based High speed four-camera vision system', *2011 9th World Congress on Intelligent Control and Automation (WCICA)*, pp. 1087–1092 [Online]. DOI: 10.1109/WCICA.2011.5970684.

Chen, X., Tian, Y., Huang, Q., Zhang, W. and Yu, Z. (2010) 'Dynamic model based ball trajectory prediction for a robot ping-pong player', *2010 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 603–608 [Online]. DOI: 10.1109/ROBIO.2010.5723394.

Cheng, X., Honda, M., Ikoma, N. and Ikenaga, T. (2016) 'Anti-occlusion observation model and automatic recovery for multi-view ball tracking in sports analysis', *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1501–1505 [Online]. DOI: 10.1109/ICASSP.2016.7471927.

Chiu, Ching-Hua, et al. (2010) 'Prediction of Ball Placement Using Computer Simulation for Wheelchair Players in Table Tennis Singles', *International Journal of Sport and Exercise Science 2.1*, pp. 1–6.

Comaniciu, D. and Meer, P. (2002) 'Mean shift: a robust approach toward feature space analysis', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619 [Online]. DOI: 10.1109/34.1000236.

Conaire, C. O., Kelly, P., Connaghan, D. and O'Connor, N. E. (2009) 'TennisSense: A platform for extracting semantic information from multi-camera tennis data', *2009 16th International Conference on Digital Signal Processing*, pp. 1–6 [Online]. DOI: 10.1109/ICDSP.2009.5201152.

Conaire, C. Ó., Kelly, P., Connaghan, D. and O'connor, N. E. (2009) 'TENNISSENSE: A PLATFORM FOR EXTRACTING SEMANTIC INFORMATION FROM MULTI-CAMERA TENNIS DATA', [Online]. Available at http://130.203.133.150/viewdoc/summary;jsessionid=5015254A3FE46E9960DBA40647751EB3?doi=10.1.1.154.9334.

Cppcheck - A tool for static C/C++ code analysis (n.d.) [Online]. Available at http://cppcheck.sourceforge.net/ (Accessed 18 February 2018).

Crabb, R., Tracey, C., Puranik, A. and Davis, J. (2008) 'Real-time foreground segmentation via range and color imaging', *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2008. CVPRW '08*, pp. 1–5 [Online]. DOI: 10.1109/CVPRW.2008.4563170.

Cui, Y., Schuon, S., Chan, D., Thrun, S. and Theobalt, C. (2010) '3D shape scanning with a time-of-flight camera', *2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1173–1180 [Online]. DOI: 10.1109/CVPR.2010.5540082.

Cui, Y., Schuon, S., Thrun, S., Stricker, D. and Theobalt, C. (2013) 'Algorithms for 3D Shape Scanning with a Depth Camera', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 5, pp. 1039–1050 [Online]. DOI: 10.1109/TPAMI.2012.190.

Cutler, R. and Davis, L. (1998) 'View-based detection and analysis of periodic motion', *Fourteenth International Conference on Pattern Recognition, 1998. Proceedings*, vol. 1, pp. 495–500 vol.1 [Online]. DOI: 10.1109/ICPR.1998.711189.

Danny Poo (2008) *Object-Oriented Programming and Java*, Springer [Online]. Available at https://www.springer.com/gp/book/9781846289620 (Accessed 14 February 2019).

Delano and L.F. (n.d.), (first) (2018) *ITTF Umpires and Referees* [Online]. Available at http://www.ittf.com/urc/courses/Umpire_Level_1_v1_files/frame.htm (Accessed 15 December 2017).

Demaj, D. (2013) *Unlocking Hawk-Eye data: What it means for tennis, the ATP, WTA and ITF. | GameSetMap* [Online]. Available at http://gamesetmap.com/?p=74 (Accessed 23 October 2013).

Dokmanic, I., Parhizkar, R., Ranieri, J. and Vetterli, M. (2015) 'Euclidean Distance Matrices: Essential theory, algorithms, and applications', *IEEE Signal Processing Magazine*, vol. 32, no. 6, pp. 12–30 [Online]. DOI: 10.1109/MSP.2015.2398954.

Dovaston, I. and Correspondent, S. N. (2012) *Goal-Line Technology Gets The Green Light* [Online]. Available at http://news.sky.com/story/956393 (Accessed 23 October 2013).

Eigenraam, D. and Rothkrantz, L. J. M. (2016) 'A smart surveillance system of distributed smart multi cameras modelled as agents', *2016 Smart Cities Symposium Prague (SCSP)*, pp. 1–6 [Online]. DOI: 10.1109/SCSP.2016.7501018.

Expert Table Tennis (2013) *Expert Table Tennis* [Online]. Available at https://www.experttabletennis.com/improve-your-table-tennis-serve/ (Accessed 18 November 2017).

Fan, L., Wang, Z., Cail, B., Tao, C., Zhang, Z., Wang, Y., Li, S., Huang, F., Fu, S. and Zhang, F. (2016) 'A survey on multiple object tracking algorithm', *2016 IEEE International Conference on Information and Automation (ICIA)*, pp. 1855–1862 [Online]. DOI: 10.1109/ICInfA.2016.7832121.

FIFA (2013) *FIFA picks GoalControl for Brazil 2014 | Sci-Tech | DW.DE | 02.04.2013* [Online]. Available at http://www.dw.de/fifa-picks-goalcontrol-for-brazil-2014/a-16714271 (Accessed 23 October 2013).

Finin, T., Fritzson, R., McKay, D. and McEntire, R. (1994) 'KQML As an Agent Communication Language', *Proceedings of the Third International Conference on Information and Knowledge Management*, CIKM '94, New York, NY, USA, ACM, pp. 456–463 [Online]. DOI: 10.1145/191246.191322 (Accessed 4 February 2018).

FIPA | IEEE (n.d.) [Online]. Available at http://www.fipa.org/ (Accessed 15 November 2017).

Fitriana, A. N., Mutijarsa, K. and Adiprawita, W. (2016) 'Color-based segmentation and feature detection for ball and goal post on mobile soccer robot game field', *2016 International Conference on Information Technology Systems and Innovation (ICITSI)*, pp. 1–4 [Online]. DOI: 10.1109/ICITSI.2016.7858232.

Fowler, S. (2012) 'How Feasible is Officiating Technology in Football?', [Online]. Available at /paper/How-Feasible-is-Officiating-Technology-in-Football-Fowler/908c7a96ffc733d4181e21da007b7240780618fc.

Gao, M., Chen, H., Zheng, S., Fang, B. and Zhang, L. (2016) 'Texture image segmentation using fused features and active contour', *2016 23rd International Conference on Pattern Recognition (ICPR)*, pp. 2036–2041 [Online]. DOI: 10.1109/ICPR.2016.7899935.

Gerven, M. van and Bohte, S. (2018) *Artificial Neural Networks as Models of Neural Information Processing*, Frontiers Media SA.

Gonzalez, R. C. and Woods, R. E. (2002) *Digital image processing*, Upper Saddle River, N.J., Prentice Hall.

Greenspan, H., Belongie, S., Goodman, R., Perona, P., Rakshit, S. and Anderson, C. H. (1994) 'Overcomplete steerable pyramid filters and rotation invariance', *1994*

Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 222–228 [Online]. DOI: 10.1109/CVPR.1994.323833.

Grzegorz J. Nalepa (2018) *Modeling with Rules Using Semantic Knowledge Engineering*, Springer [Online]. Available at https://www.springer.com/gp/book/9783319666549 (Accessed 14 February 2019).

Haralick, R. M., Shanmugam, K. and Dinstein, I. (1973) 'Textural Features for Image Classification', *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-3, no. 6, pp. 610–621 [Online]. DOI: 10.1109/TSMC.1973.4309314.

Hawk - Eye (2017) *Tennis: Hawk-Eye* [Online]. Available at http://www.hawkeyeinnovations.co.uk/page/sports-officiating/tennis (Accessed 23 October 2017).

Hawk-Eye (2018) *Goal-line technology (GLT)* [Online]. Available at http://www.hawkeyeinnovations.co.uk/page/sports-officiating/football (Accessed 29 October 2017).

Hawk-Eye Innovations (2018) *Hawk-Eye Innovations* [Online]. Available at https://www.hawkeyeinnovations.com/ (Accessed 18 June 2018).

Hopgood, A. A. (2012) *Intelligent Systems for Engineers and Scientists, Third Edition*, 3 edition., New Haven, CT, CRC Press.

Hossein-Khani, J., Soltanian-Zadeh, H., Kamarei, M. and Staadt, O. (2011) 'Ball Detection with the Aim of Corner Event Detection in Soccer Video', *2011 IEEE Ninth International Symposium on Parallel and Distributed Processing with Applications Workshops*, pp. 147–152 [Online]. DOI: 10.1109/ISPAW.2011.41.

Hsin, C., Le, H.-N. and Shin, S.-J. (2011) 'Color to grayscale transform preserving natural order of hues', *2011 International Conference on Electrical Engineering and Informatics (ICEEI)*, pp. 1–6 [Online]. DOI: 10.1109/ICEEI.2011.6021794.

Huang, C. l, Shih, H. c and Chen, C. l (2006) 'Shot and Scoring Events Identification of Basketball Videos', *2006 IEEE International Conference on Multimedia and Expo*, pp. 1885–1888 [Online]. DOI: 10.1109/ICME.2006.262923.

Huang, Q., Cox, S., Zhou, X. and Xie, L. (2012) 'Detection of ball hits in a tennis game using audio and visual information', *Signal Information Processing Association Annual Summit and Conference (APSIPA ASC), 2012 Asia-Pacific*, pp. 1–10.

Hyndman, R. J. and Koehler, A. B. (2006) 'Another look at measures of forecast accuracy', *International Journal of Forecasting*, vol. 22, no. 4, pp. 679–688 [Online]. DOI: 10.1016/j.ijforecast.2006.03.001.

International Table Tennis Federation (2018) *ITTF Handbook* [Online]. Available at https://www.ittf.com/handbook/ (Accessed 14 September 2017).

Ishii, I., Koike, T., Gao, H. and Takaki, T. (2011) 'Fast 3D shape measurement using structured light projection for a one-directionally moving object', *IECON 2011 - 37th*

Annual Conference on IEEE Industrial Electronics Society, pp. 135–140 [Online]. DOI: 10.1109/IECON.2011.6119301.

ITTF, 2018 (n.d.) *International Table Tennis Federation* [Online]. Available at https://www.ittf.com/handbook/ (Accessed 6 April 2018).

Jin, T.-S., Morioka, K. and Hashimoto, H. (2006) 'Distributed Sensor Network for Multi-Agent Motion Tracking in Intelligent Space', *SICE-ICASE, 2006. International Joint Conference*, pp. 3716–3721 [Online]. DOI: 10.1109/SICE.2006.315169.

Kale, K., Pawar, S. and Dhulekar, P. (2015) 'Moving object tracking using optical flow and motion vector estimation', *2015 4th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO) (Trends and Future Directions)*, pp. 1–6 [Online]. DOI: 10.1109/ICRITO.2015.7359323.

Kamarposhty, M. S., Asadollahi, H. and Teymoori, M. M. (2009) 'Line Detection in Sport Images Using Multi-agent Systems', *2009 International Conference on Future Computer and Communication*, pp. 319–323 [Online]. DOI: 10.1109/ICFCC.2009.61.

Katalenic, A., Draganjac, I., Mutka, A. and Bogdan, S. (2009) 'Fast visual tracking and localization in multi-agent systems', *4th IEEE Conference on Industrial Electronics and Applications, 2009. ICIEA 2009*, pp. 1864–1870 [Online]. DOI: 10.1109/ICIEA.2009.5138527.

Kim, J. S. and Kim, M. G. (2017) 'A new ICF-based ball tracking system with multi-exposure cameras for virtual sports', *2017 19th International Conference on Advanced Communication Technology (ICACT)*, pp. 640–643 [Online]. DOI: 10.23919/ICACT.2017.7890170.

Kim, J. Y. and Kim, T. Y. (2009) 'Soccer Ball Tracking Using Dynamic Kalman Filter with Velocity Control', *Imaging and Visualization 2009 Sixth International Conference on Computer Graphics*, pp. 367–374 [Online]. DOI: 10.1109/CGIV.2009.87.

Kohei Arai (2018) *Intelligent Computing*, Springer, vol. 1 [Online]. Available at https://www.springer.com/gp/book/9783030011734 (Accessed 14 February 2019).

Kolkur, S. and Kalbande, D. R. (2016) 'Survey of texture based feature extraction for skin disease detection', *2016 International Conference on ICT in Business Industry Government (ICTBIG)*, pp. 1–6 [Online]. DOI: 10.1109/ICTBIG.2016.7892649.

Lee, K., Lee, C., Kim, S.-A. and Kim, Y.-H. (2012) 'Fast object detection based on color histograms and local binary patterns', *TENCON 2012 - 2012 IEEE Region 10 Conference*, pp. 1–4 [Online]. DOI: 10.1109/TENCON.2012.6412323.

Leo, M., Mosca, N., Spagnolo, P., Mazzeo, P. L. and Distante, A. (2008) 'Real-time multi-view event detection in soccer games', *2008 Second ACM/IEEE International Conference on Distributed Smart Cameras*, pp. 1–10 [Online]. DOI: 10.1109/ICDSC.2008.4635729.

Li, Yingzhu, Shark, L.-K., Hobbs, S. J. and Ingham, J. (2010) 'Real-Time Immersive Table Tennis Game for Two Players with Motion Tracking', *Information Visualisation (IV), 2010 14th International Conference*, pp. 500–505 [Online]. DOI: 10.1109/IV.2010.97.

Li, Yongping, Wang, Y., Qi, Y. and Li, H. (2010) 'Multi-agent based particle filter for moving object tracking', *2010 International Conference on Computer Application and System Modeling (ICCASM)*, vol. 4, pp. V4-124-V4-128 [Online]. DOI: 10.1109/ICCASM.2010.5619310.

Linderoth, M., Robertsson, A. and Johansson, R. (2013) 'Color-based detection robust to varying illumination spectrum', *2013 IEEE Workshop on Robot Vision (WORV)*, pp. 120–125 [Online]. DOI: 10.1109/WORV.2013.6521924.

Liu, J., Fang, Z., Zhang, K. and Tan, M. (2014) 'Improved high-speed vision system for table tennis robot', *2014 IEEE International Conference on Mechatronics and Automation (ICMA)*, pp. 652–657 [Online]. DOI: 10.1109/ICMA.2014.6885774.

Lu, X. and Yang, Z. (2017) 'A nonlinear background updating scheme', *2017 IEEE 26th International Symposium on Industrial Electronics (ISIE)*, pp. 1305–1310 [Online]. DOI: 10.1109/ISIE.2017.8001434.

Ma, C., Gao, W., Yang, L. and Liu, Z. (2010) 'An improved Sobel algorithm based on median filter', *2010 2nd International Conference on Mechanical and Electronics Engineering*, vol. 1, pp. V1-88-V1-92 [Online]. DOI: 10.1109/ICMEE.2010.5558590.

Maddalena, L. and Petrosino, A. (2013) 'Stopped Object Detection by Learning Foreground Model in Videos', *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 5, pp. 723–735 [Online]. DOI: 10.1109/TNNLS.2013.2242092.

Mahmood, T., Ahmed, S. O., Swaleh, M. H. and Nayyer, S. H. (2011) 'A-Eye: Automating the Role of the Third Umpire in the Game of Cricket', *2011 International Conference on Information Science and Applications (ICISA)*, pp. 1–11 [Online]. DOI: 10.1109/ICISA.2011.5772364.

Maksai, A., Wang, X. and Fua, P. (2016) 'What Players do with the Ball: A Physically Constrained Interaction Modeling', *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 972–981 [Online]. DOI: 10.1109/CVPR.2016.111.

Mallat, S. G. (1989) 'A theory for multiresolution signal decomposition: the wavelet representation', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 7, pp. 674–693 [Online]. DOI: 10.1109/34.192463.

Margaria, T. and Steffen, B. (2016) *Leveraging Applications of Formal Methods, Verification and Validation: Discussion, Dissemination, Applications: 7th International Symposium, ISoLA 2016, Imperial, Corfu, Greece, October 10-14, 2016, Proceedings*, Springer.

MathWorks, 2018 (n.d.) *MATLAB* [Online]. Available at https://uk.mathworks.com/products/matlab.html (Accessed 18 February 2018).

McIlroy, P. (2008) ‘Hawk-Eye: Augmented reality in sports broadcasting and officiating’, *7th IEEE/ACM International Symposium on Mixed and Augmented Reality, 2008. ISMAR 2008*, pp. xiv–xiv [Online]. DOI: 10.1109/ISMAR.2008.4637309.

McNeill, H., Salim, Y., Ayllon, M. and Goncer, T. (2016) ‘Design of a decision support system for safe landing of high-drag, low-inertia light sport and experimental aircraft’, *2016 IEEE Systems and Information Engineering Design Symposium (SIEDS)*, pp. 142–147 [Online]. DOI: 10.1109/SIEDS.2016.7489287.

Mitchell, M. (1998) *An Introduction to Genetic Algorithms*, MIT Press.

Mukai, R., Asano, T. and Hara, H. (2011) ‘Analysis and evaluation of tennis plays by computer vision’, *2011 International Conference on Mechatronics and Automation (ICMA)*, pp. 784–788 [Online]. DOI: 10.1109/ICMA.2011.5985761.

Myint, H., Wong, P., Dooley, L. and Hopgood, A. (2015) ‘Tracking a table tennis ball for umpiring purposes’, *2015 14th IAPR International Conference on Machine Vision Applications (MVA)*, pp. 170–173 [Online]. DOI: 10.1109/MVA.2015.7153160.

Myint, Hnin (2016) ‘Tracking a table tennis ball for umpiring purposes using a multi-agent system’, *The 20th International Conference on Image Processing, Computer Vision, & Pattern Recognition, Las Vegas, USA*.

Nummiaro, K., Koller-Meier, E. and Gool, L. V. (2003) ‘An adaptive color-based particle filter’, *Image and Vision Computing*, vol. 21, no. 1, pp. 99–110 [Online]. DOI: 10.1016/S0262-8856(02)00129-4.

Oklobdzija, V. G. (2001) *The Computer Engineering Handbook*, CRC Press.

OpenCV library 2017 (n.d.) [Online]. Available at https://opencv.org/ (Accessed 17 November 2017).

ordo.open.ac.uk (2019) *Multi-view Table Tennis Ball Tracking Results* [Online]. Available at https://ordo.open.ac.uk/articles/Multi-view_Table_Tennis_Ball_Tracking_Results/7746203 (Accessed 27 February 2019).

Owens, N., Harris, C. and Stennett, C. (2003) ‘Hawk-eye tennis system’, *International Conference on Visual Information Engineering, 2003. VIE 2003*, pp. 182–185 [Online]. DOI: 10.1049/cp:20030517.

Pan, M. K. X. J. and Niemeyer, G. (2017) ‘Catching a real ball in virtual reality’, *2017 IEEE Virtual Reality (VR)*, pp. 269–270 [Online]. DOI: 10.1109/VR.2017.7892280.

Parekh, H. S., Thakore, D. G. and Jaliya, U. K. (2007) ‘A Survey on Object Detection and Tracking Methods’, vol. 2, no. 2, p. 9.

Pingali, G., Opalach, A. and Jean, Y. (2000) ‘Ball tracking and virtual replays for innovative tennis broadcasts’, vol. 4, pp. 152–156 vol.4 [Online]. DOI: 10.1109/ICPR.2000.902885.

Pinho, R. R., Manuel, J., Tavares, R. S. and Correia, M. V. (2006) 'Efficient Approximation of the Mahalanobis Distance for Tracking with the Kalman Filter', *in CompIMAGE - Computational Modelling of Objects Represented in Images: Fundamentals, Methods and Applications*, pp. 84–92.

Poslad, S. (2007) 'Specifying Protocols for Multi-agent Systems Interaction', *ACM Trans. Auton. Adapt. Syst.*, vol. 2, no. 4 [Online]. DOI: 10.1145/1293731.1293735 (Accessed 4 February 2018).

Putri, D. C., Christie, D. A. and Musa, P. (2017) 'Robust ball color auto-calibration for tracking', *2017 Second International Conference on Informatics and Computing (ICIC)*, pp. 1–5 [Online]. DOI: 10.1109/IAC.2017.8280624.

Qazi, T., Mukherjee, P., Srivastava, S., Lall, B. and Chauhan, N. R. (2015) 'Automated ball tracking in tennis videos', *2015 Third International Conference on Image Information Processing (ICIIP)*, pp. 236–240 [Online]. DOI: 10.1109/ICIIP.2015.7414772.

R. E. Kalman, T. (2001) 'A New Approach to Linear Filtering and Prediction Problems', in *Control Theory:Twenty-Five Seminal Papers (1932-1981)*, Wiley-IEEE Press, pp. 167–179 [Online]. DOI: 10.1109/9780470544334.ch9 (Accessed 30 October 2017).

Ratnayake, K. and Amer, M. A. (2015) 'Object tracking with adaptive motion modeling of particle filter and support vector machines', *2015 IEEE International Conference on Image Processing (ICIP)*, pp. 1140–1144 [Online]. DOI: 10.1109/ICIP.2015.7350978.

Reddy, K. R., Priya, K. H. and Neelima, N. (2015) 'Object Detection and Tracking – A Survey', *2015 International Conference on Computational Intelligence and Communication Networks (CICN)*, pp. 418–421 [Online]. DOI: 10.1109/CICN.2015.317.

Rusdorf, S., Brunnett, G., Lorenz, M. and Winkler, T. (2007) 'Real-Time Interaction with a Humanoid Avatar in an Immersive Table Tennis Simulation', *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 1, pp. 15–25 [Online]. DOI: 10.1109/TVCG.2007.18.

Seif, A., Salut, M. M. and Marsono, M. N. (2010) 'A hardware architecture of Prewitt edge detection', *2010 IEEE Conference on Sustainable Utilization and Development in Engineering and Technology*, pp. 99–101 [Online]. DOI: 10.1109/STUDENT.2010.5686999.

Sengar, S. S. and Mukhopadhyay, S. (2016) 'A novel method for moving object detection based on block based frame differencing', *2016 3rd International Conference on Recent Advances in Information Technology (RAIT)*, pp. 467–472 [Online]. DOI: 10.1109/RAIT.2016.7507946.

Seo, B. K. and Wuest, H. (2016) 'Robust 3D Object Tracking Using an Elaborate Motion Model', *2016 IEEE International Symposium on Mixed and Augmented Reality (ISMAR-Adjunct)*, pp. 70–71 [Online]. DOI: 10.1109/ISMAR-Adjunct.2016.0042.

Shahjalal, M. A., Ahmad, Z., Rayan, R. and Alam, L. (2017) 'An approach to automate the scorecard in cricket with computer vision and machine learning', *2017 3rd International Conference on Electrical Information and Communication Technology (EICT)*, pp. 1–6 [Online]. DOI: 10.1109/EICT.2017.8275204.

Shao, L. and Xie, G. (2012) 'Real-time tracking of moving objects on a water surface', *2012 International Conference on Mechatronics and Automation (ICMA)*, pp. 2114–2119 [Online]. DOI: 10.1109/ICMA.2012.6285670.

Sheu, M.-H., Tsai, W.-K., Hu, C.-C. and Tsao, C.-H. (2010) 'Fast Texture-Based Object Tracking Algorithm on Embedded Platform', *2010 Fifth International Conference on Frontier of Computer Science and Technology (FCST)*, pp. 511–514 [Online]. DOI: 10.1109/FCST.2010.27.

Shih, H. C. (2017) 'A Survey on Content-aware Video Analysis for Sports', *IEEE Transactions on Circuits and Systems for Video Technology*, vol. PP, no. 99, pp. 1–1 [Online]. DOI: 10.1109/TCSVT.2017.2655624.

Singh, R., Godse, D. M. and Biradar, T. D. (2013) 'Comparison of Different Video Object Tracking Methods', *International Journal of Advanced Research in Computer Science and Software Engineering*, p. 6.

Singh, S. and Singh, R. (2015) 'Comparison of various edge detection techniques', *2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom)*, pp. 393–396.

Sports live production (2013) *Hawk-Eye Wins Contract for Premier League Goal-Line Technology | LIVE-PRODUCTION.TV* [Online]. Available at http://www.live-production.tv/news/sports/hawk-eye-wins-contract-premier-league-goal-line-technology.html (Accessed 23 October 2013).

Srivastav, N., Agrwal, S. L., Gupta, S. K., Srivastava, S. R., Chacko, B. and Sharma, H. (2017) 'Hybrid object detection using improved three frame differencing and background subtraction', *2017 7th International Conference on Cloud Computing, Data Science Engineering - Confluence*, pp. 613–617 [Online]. DOI: 10.1109/CONFLUENCE.2017.7943225.

Sudhir, G., Lee, J. C. M. and Jain, A. K. (1998) 'Automatic classification of tennis video for high-level content-based retrieval', *, 1998 IEEE International Workshop on Content-Based Access of Image and Video Database, 1998. Proceedings*, pp. 81–90 [Online]. DOI: 10.1109/CAIVD.1998.646036.

Sun, C. and Lam, K.-M. (2013) 'Multiple-Kernel, Multiple-Instance Similarity Features for Efficient Visual Object Detection', *IEEE Transactions on Image Processing*, vol. 22, no. 8, pp. 3050–3061 [Online]. DOI: 10.1109/TIP.2013.2255303.

Swain, M. J. and Ballard, D. H. (1991) 'Color indexing', *International Journal of Computer Vision*, vol. 7, no. 1, pp. 11–32 [Online]. DOI: 10.1007/BF00130487.

Swears, E. and Hoogs, A. (2012) 'Learning and recognizing complex multi-agent activities with applications to american football plays', *2012 IEEE Workshop on*

Applications of Computer Vision (WACV), pp. 409–416 [Online]. DOI: 10.1109/WACV.2012.6163027.

Table Tennis Master (2017) *All About Ping Pong Balls* [Online]. Available at http://www.tabletennismaster.com/page/ping-pong-balls (Accessed 7 September 2017).

Takahashi, M., Ikeya, K., Kano, M., Ookubo, H. and Mishina, T. (2016) 'Robust volleyball tracking system using multi-view cameras', *2016 23rd International Conference on Pattern Recognition (ICPR)*, pp. 2740–2745 [Online]. DOI: 10.1109/ICPR.2016.7900050.

Takahashi, M., Yamanouchi, Y. and Nakamura, T. (2015) 'Real-Time Ball Position Measurement for Football Games Based on Ball's Appearance and Motion Features', *2015 11th International Conference on Signal-Image Technology Internet-Based Systems (SITIS)*, pp. 287–294 [Online]. DOI: 10.1109/SITIS.2015.46.

Tamaki, T., Wang, H., Raytchev, B., Kaneda, K. and Ushiyama, Y. (2012) 'Estimating the spin of a table tennis ball using Inverse Compositional Image Alignment', *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1457–1460 [Online]. DOI: 10.1109/ICASSP.2012.6288166.

Tanaka, T. and Miura, Y. (2017) 'A study on inter-frame differencing for compressed moving images', *2017 IEEE International Conference on Consumer Electronics - Taiwan (ICCE-TW)*, pp. 13–14 [Online]. DOI: 10.1109/ICCE-China.2017.7990971.

Tang, H., Shi, Y., Xia, J. and Yin, H. (2008) 'A fast and accurate boundary tracking of moving objects in video', *International Conference on Information and Automation, 2008. ICIA 2008*, pp. 681–685 [Online]. DOI: 10.1109/ICINFA.2008.4608085.

Teachabarikiti, K., Chalidabhongse, T. H. and Thammano, A. (2010) 'Players tracking and ball detection for an automatic tennis video annotation', *2010 11th International Conference on Control Automation Robotics Vision (ICARCV)*, pp. 2461–2494 [Online]. DOI: 10.1109/ICARCV.2010.5707906.

Tong-yao, W. (2011) 'The motion detection based on background difference method and active contour model', *Information Technology and Artificial Intelligence Conference (ITAIC), 2011 6th IEEE Joint International*, vol. 2, pp. 480–483 [Online]. DOI: 10.1109/ITAIC.2011.6030378.

Toyama, K., Krumm, J., Brumitt, B. and Meyers, B. (1999) 'Wallflower: principles and practice of background maintenance', *The Proceedings of the Seventh IEEE International Conference on Computer Vision, 1999*, vol. 1, pp. 255–261 vol.1 [Online]. DOI: 10.1109/ICCV.1999.791228.

Tsang, J. (2013) *Technology in Tennis: Hawk-eye* [Online]. Available at http://blog.jasontsang.ca/2006/07/technology-in-tennis-hawk-eye.html (Accessed 23 October 2013).

Wallace, D. R., Ippolito, L. M. and Cuthill, B. B. (1996) *Reference Information for the Software Verification and Validation Process*, DIANE Publishing.

Wang, J. R. and Parameswaran, N. (2004) 'Survey of sports video analysis: research issues and applications', *Proceedings of the Pan-Sydney area workshop on Visual information processing*, VIP '05, Darlinghurst, Australia, Australia, Australian Computer Society, Inc., pp. 87–90 [Online]. Available at http://dl.acm.org/citation.cfm?id=1082121.1082136 (Accessed 31 July 2013).

Wang, Q., Zhang, X. and Xu, D. (2012) 'Human behavior imitation for a robot to play table tennis', *Control and Decision Conference (CCDC), 2012 24th Chinese*, pp. 1482–1487 [Online]. DOI: 10.1109/CCDC.2012.6243007.

Wang, Y., Cheng, X., Ikoma, N., Honda, M. and Ikenaga, T. (2016) 'Motion Prejudgment Dependent Mixture System Noise in System Model for Tennis Ball 3D Position Tracking by Particle Filter', *2016 Joint 8th International Conference on Soft Computing and Intelligent Systems (SCIS) and 17th International Symposium on Advanced Intelligent Systems (ISIS)*, pp. 124–129 [Online]. DOI: 10.1109/SCIS-ISIS.2016.0038.

Wei, X., Lucey, P., Morgan, S. and Sridharan, S. (2016) 'Forecasting the Next Shot Location in Tennis Using Fine-Grained Spatiotemporal Tracking Data', *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 11, pp. 2988–2997 [Online]. DOI: 10.1109/TKDE.2016.2594787.

Wong, K. C. P. and Dooley, L. S. (2010) 'High-motion table tennis ball tracking for umpiring applications', *2010 IEEE 10th International Conference on Signal Processing (ICSP)*, pp. 2460–2463 [Online]. DOI: 10.1109/ICOSP.2010.5657001.

Wong, K. C. P. and Dooley, L. S. (2017) *XGMT | Open University Table Tennis Video Database* [Online]. Available at http://users.mct.open.ac.uk/xgmt/outtdb/ (Accessed 13 September 2017).

Wong, P. (2009) 'Identifying table tennis balls from real match scenes using image processing and artificial intelligence techniques', *International Journal of Simulation Systems, Science & Technology*, vol. 10, pp. 6–14.

Wong, P. and Dooley, L. (2011) 'Tracking Table Tennis Balls in Real Match Scenes for Umpiring Applications', *British Journal of Mathematics & Computer Science*, vol. 1, pp. 228–241.

Wong, P. K. C. (2007) 'Developing an Intelligent Assistant for Table Tennis Umpires', *First Asia International Conference on Modelling Simulation, 2007. AMS '07*, pp. 340–345 [Online]. DOI: 10.1109/AMS.2007.32.

Wong, P. K. C. (2008) 'Developing an Intelligent Table Tennis Umpiring System: Identifying the Ball from the Scene', *Second Asia International Conference on Modeling Simulation, 2008. AICMS 08*, pp. 445–450 [Online]. DOI: 10.1109/AMS.2008.110.

Wooldridge, M. (2008) *An Introduction to MultiAgent Systems*, John Wiley & Sons.

Wooldridge, M. J. and Jennings, N. (1995) *Intelligent Agents*, Springer-Verlag.

Wormell, D. and Foxlin, E. (2003) 'Advancements in 3D interactive devices for virtual environments', *Proceedings of the workshop on Virtual environments 2003*, EGVE '03, New York, NY, USA, ACM, pp. 47–56 [Online]. DOI: 10.1145/769953.769959 (Accessed 15 September 2013).

Wu, Y., He, X. and Nguyen, T. Q. (2017) 'Moving Object Detection With a Freely Moving Camera via Background Motion Subtraction', *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 2, pp. 236–248 [Online]. DOI: 10.1109/TCSVT.2015.2493499.

Xiao, C. and Yilmaz, A. (2016) 'Efficient tracking with distinctive target colors and silhouette', *2016 23rd International Conference on Pattern Recognition (ICPR)*, pp. 2728–2733 [Online]. DOI: 10.1109/ICPR.2016.7900048.

Xu, Z., Zhu, S. and Cheng, Y. (2017) 'Object detection via superpixel and 3D self-organizing background subtraction', *2017 29th Chinese Control And Decision Conference (CCDC)*, pp. 1754–1759 [Online]. DOI: 10.1109/CCDC.2017.7978800.

Yadav, D. K., Sharma, L. and Bharti, S. K. (2014) 'Moving object detection in real-time visual surveillance using background subtraction technique', *2014 14th International Conference on Hybrid Intelligent Systems*, pp. 79–84 [Online]. DOI: 10.1109/HIS.2014.7086176.

Yan, F., Christmas, W. and Kittler, J. (2005) 'A tennis ball tracking algorithm for automatic annotation of tennis match', *In BMVC*, pp. 619–628.

Yilmaz, A., Javed, O. and Shah, M. (2006) 'Object Tracking: A Survey', *ACM Computing Surveys*, vol. 38, no. 4 [Online]. DOI: 10.1145/1177352.1177355 (Accessed 24 June 2018).

Yu, X., Sim, C.-H., Wang, J. R. and Cheong, L.-F. (2004) 'A trajectory-based ball detection and tracking algorithm in broadcast tennis video', *2004 International Conference on Image Processing, 2004. ICIP '04*, vol. 2, pp. 1049-1052 Vol.2 [Online]. DOI: 10.1109/ICIP.2004.1419482.

Zaveri, M. A., Merchant, S. N. and Desai, U. B. (2004) 'Small and fast moving object detection and tracking in sports video sequences', *2004 IEEE International Conference on Multimedia and Expo, 2004. ICME '04*, vol. 3, pp. 1539-1542 Vol.3 [Online]. DOI: 10.1109/ICME.2004.1394540.

Zhang, B., Dou, W. and Chen, L. (2006) 'Ball Hit Detection in Table Tennis Games Based on Audio Analysis', *18th International Conference on Pattern Recognition, 2006. ICPR 2006*, vol. 3, pp. 220–223 [Online]. DOI: 10.1109/ICPR.2006.314.

Zhang, S., Lin, W., Lu, P., Li, W. and Deng, S. (2017) 'Kill two birds with one stone: Boosting both object detection accuracy and speed with adaptive patch-of-interest composition', *2017 IEEE International Conference on Multimedia Expo Workshops (ICMEW)*, pp. 447–452 [Online]. DOI: 10.1109/ICMEW.2017.8026235.

Zhang, Z. (2000) ‘A flexible new technique for camera calibration’, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334 [Online]. DOI: 10.1109/34.888718.

Zhang, Z., Xu, D. and Tan, M. (2010) ‘Visual Measurement and Prediction of Ball Trajectory for Table Tennis Robot’, *IEEE Transactions on Instrumentation and Measurement*, vol. 59, no. 12, pp. 3195–3205 [Online]. DOI: 10.1109/TIM.2010.2047128.

Zhao, S., Zhang, S. and Zhang, L. (2017) ‘Towards Occlusion Handling: Object Tracking With Background Estimation’, *IEEE Transactions on Cybernetics*, vol. PP, no. 99, pp. 1–15 [Online]. DOI: 10.1109/TCYB.2017.2727138.

Zheng, S., Qiao, H., Jia, L. and Fukuda, T. (2012) ‘A simplified minimum enclosing ball based fast incremental support vector machine (SVM) algorithm for person detection and tracking’, *Proceedings of the 10th World Congress on Intelligent Control and Automation*, pp. 4936–4941 [Online]. DOI: 10.1109/WCICA.2012.6359413.

Zhou, X., Huang, Q., Xie, L. and Cox, S. (2013) ‘A two layered data association approach for ball tracking’, *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 2317–2321 [Online]. DOI: 10.1109/ICASSP.2013.6638068.

Zhou, X., Xie, L., Huang, Q., Cox, S. J. and Zhang, Y. (2015) ‘Tennis Ball Tracking Using a Two-Layered Data Association Approach’, *IEEE Transactions on Multimedia*, vol. 17, no. 2, pp. 145–156 [Online]. DOI: 10.1109/TMM.2014.2380914.

Ziółko, B., Emms, D. and Ziółko, M. (2017) ‘Fuzzy Evaluations of Image Segmentations’, *IEEE Transactions on Fuzzy Systems*, vol. PP, no. 99, pp. 1–1 [Online]. DOI: 10.1109/TFUZZ.2017.2752130.

Zivkovic, Z. (2004) ‘Improved adaptive Gaussian mixture model for background subtraction’, *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004*, vol. 2, pp. 28-31 Vol.2 [Online]. DOI: 10.1109/ICPR.2004.1333992.